User Manual

Allen-Bradley

Micro830 and Micro850 Programmable Controllers

Catalog Numbers Bulletin 2080-LC30 and 2080-LC50

Important User Information

Solid-state equipment has operational characteristics differing from those of electromechanical equipment. Safety Guidelines for the Application, Installation and Maintenance of Solid State Controls (publication SGI-1.1 available from your local Rockwell Automation sales office or online at http://www.rockwellautomation.com/literature/) describes some important differences between solid-state equipment and hard-wired electromechanical devices. Because of this difference, and also because of the wide variety of uses for solid-state equipment, all persons responsible for applying this equipment must satisfy themselves that each intended application of this equipment is acceptable.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

	WARNING: Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.
	ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence
	SHOCK HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.
	BURN HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.
	ARC FLASH HAZARD: Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).
IMPORTANT	Identifies information that is critical for successful application and understanding of the product.

Allen-Bradley, Rockwell Software, Rockwell Automation, Micro800, Micro830, Micro850, Connected Components Workbench, and TechConnect are trademarks of Rockwell Automation, Inc.

Trademarks not belonging to Rockwell Automation are property of their respective companies.

Read this preface to familiarize yourself with the rest of the manual. It provides information concerning:

- who should use this manual
- the purpose of this manual
- related documentation
- supporting information for Micro800™

Who Should Use this Manual

Use this manual if you are responsible for designing, installing, programming, or troubleshooting control systems that use Micro800 controllers.

You should have a basic understanding of electrical circuitry and familiarity with relay logic. If you do not, obtain the proper training before using this product.

Purpose of this Manual

This manual is a reference guide for Micro800 controllers, plug-in modules and accessories. It describes the procedures you use to install, wire, and troubleshoot your controller. This manual:

- explains how to install and wire your controllers
- gives you an overview of the Micro800 controller system

Refer to the Online Help provided with Connected Components Workbench™ software for more information on programming your Micro800 controller.

Additional Resources

These documents contain additional information concerning related Rockwell Automation products.

Resource	Description
Micro800 Analog and Discrete Expansion I/O Modules 2080-UM003	Information on features, configuration, wiring, installation, and specifications for the Micro800 expansion I/O modules.
Micro800 Plug-in Modules 2080-UM004	Information on features, configuration, installation, wiring, and specifications for the Micro800 plug-in modules.
Micro800 Programmable Controllers General Instructions 2080-RM001	Information on instruction sets for developing programs for use in Micro800 control systems.
Micro800 Programmable Controllers: Getting Started with Motion Control Using a Simulated Axis 2080-QS001	Provides quickstart instructions for implementing a motion control project in Connected Components Workbench software.
Micro800 Programmable Controllers: Getting Started with CIP Client Messaging 2080-QS002	Provides quickstart instructions for using CIP GENERIC and CIP Symbolic Messaging.
Micro800 Programmable Controllers: Getting Started with PanelView Plus 2080-QS003	Provides quickstart instructions for using global variables for Micro800 controllers together with PanelView Plus HMI terminals.
Configuring Micro800 Controllers on FactoryTalk Gateway 2080-QS005	Provides quickstart instructions for configuring a Micro800 controller on FactoryTalk Gateway.

Resource	Description
Kinetix 3 Motion Control Indexing Application Connected Components Accel Toolkit Quick Start CC-QS025	Provides quick start instructions for implementing a Kinetix 3 drive indexing application using Connected Components Workbench software and a Micro800 controller.
Motion Control PTO Application Building Block CC-QS033	Provides quick start instructions for implementing PTO motion control of a Kinetix 3 drive using Connected Components Workbench software and a Micro800 controller.
Micro800 Programmable Controller External AC Power Supply Installation Instructions 2080-IN001	Information on mounting and wiring the optional external power supply.
Micro830 Programmable Controllers Installation Instructions 2080-IN002	Information on mounting and wiring the Micro830 10-point Controllers.
Micro830 Programmable Controllers Installation Instructions 2080-IN003	Information on mounting and wiring the Micro830 16-point Controllers.
Micro830 Programmable Controllers Installation Instructions 2080-IN004	Information on mounting and wiring the Micro830 24-point Controllers.
Micro830 Programmable Controllers Installation Instructions 2080-IN005	Information on mounting and wiring the Micro830 48-point Controllers.
Micro850 Programmable Controllers Installation Instructions 2080-IN007	Information on mounting and wiring the Micro850 24-point Controllers
Micro850 Programmable Controllers Installation Instructions 2080-IN008	Information on mounting and wiring the Micro850 48-point Controllers
Micro800 16-point and 32-point 12/24V Sink/ Source Input Modules Installation Instructions 2085-IN001	Information on mounting and wiring the expansion I/O modules (2085-IQ16, 2085-IQ32T)
Micro800 Bus Terminator Module Installation Instruction 2085-IN002	Information on mounting and wiring the expansion I/O bus terminator (2085-ECR)
Micro800 16-Point Sink and 16-Point Source 12/ 24V DC Output Modules Installation Instructions 2085-IN003	Information on mounting and wiring the expansion I/O modules (2085-OV16, 2085-OB16)
Micro800 8-Point and 16-Point AC/DC Relay Output Modules Installation Instructions 2085-IN004	Information on mounting and wiring the expansion I/O modules (2085-OW8, 2085-OW16)
Micro800 8-Point Input and 8-Point Output AC Modules Installation Instructions 2085-IN005	Information on mounting and wiring the expansion I/O modules (2085-IA8, 2085-IM8, 2085-OA8)
Micro800 4-channel and 8-channel Analog Voltage/current Input and Output Modules Installation Instructions 2085-IN006	Information on mounting and wiring the expansion I/O modules (2085-IF4, 2085-IF8, 2085-OF4)
Micro800 4-channel Thermocouple/RTD Input Module 2085-IN007	Information on mounting and wiring the expansion I/O module (2085-IRT4)
Micro800 RS232/485 Isolated Serial Port Plug-in Module Wiring Diagrams 2080-WD002	Information on mounting and wiring the Micro800 RS232/485 Isolated Serial Port Plug-in Module.
Micro800 Non-isolated Unipolar Analog Input Plug-in Module Wiring Diagrams 2080-WD003	Information on mounting and wiring the Micro800 Non-isolated Unipolar Analog Input Plug-in Module.
Micro800 Non-isolated Unipolar Analog Output Plug-in Module Wiring Diagrams 2080-WD004	Information on mounting and wiring the Micro800 Non-isolated Unipolar Analog Output Plug-in Module.
Micro800 Non-isolated RTD Plug-in Module Wiring Diagrams 2080-WD005	Information on mounting and wiring the Micro800 Non-isolated RTD Plug-in Module.

Resource	Description
Micro800 Non-isolated Thermocouple Plug-in Module Wiring Diagrams 2080-WD006	Information on mounting and wiring the Micro800 Non-isolated Thermocouple Plug-in Module.
Micro800 Memory Backup and High Accuracy RTC Plug-In Module Wiring Diagrams 2080-WD007	Information on mounting and wiring the Micro800 Memory Backup and High Accuracy RTC Plug-In Module.
Micro800 6-Channel Trimpot Analog Input Plug-In Module Wiring Diagrams 2080-WD008	Information on mounting and wiring the Micro800 6-Channel Trimpot Analog Input Plug-In Module.
Micro800 Digital Relay Output Plug-in Module Wiring Diagrams 2080-WD010	Information on mounting and wiring the Micro800 Digital Relay Output Plug-in Module.
Micro800 Digital Input, Output, and Combination Plug-in Modules Wiring Diagrams 2080-WD011	Information on mounting and wiring the Micro800 Digital Input, Output, and Combination Plug-in Modules.
Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1	Provides general guidelines for installing a Rockwell Automation industrial system.
Product Certifications website, http://www.rockwellautomation.com/products/certification/	Provides declarations of conformity, certificates, and other certification details.
Application Considerations for Solid-State Controls SGI-1.1	A description of important differences between solid-state programmable controller products and hard-wired electromechanical devices.
National Electrical Code - Published by the National Fire Protection Association of Boston, MA.	An article on wire sizes and types for grounding electrical equipment.
Allen-Bradley Industrial Automation Glossary AG-7.1	A glossary of industrial automation terms and abbreviations.

You can view or download publications at http://www.rockwellautomation.com/literature/. To order paper copies of technical documentation, contact your local Rockwell Automation distributor or sales representative.

You can download the latest version of Connected Components Workbench for your Micro800 at the URL below.

http://www.rockwellautomation.com/global/products-technologies/connected-components/.

Notes:

Preface

Who Should Use this Manual ... iii
Purpose of this Manual ... iii
Additional Resources ... iii

Chapter 1

Hardware Overview

Hardware Features ... 1
- Micro830 Controllers ... 2
- Micro850 Controllers ... 4
- Programming Cables ... 6
- Embedded Serial Port Cables ... 7
- Embedded Ethernet Support ... 7

Chapter 2

About Your Controller

Programming Software for Micro800 Controllers ... 9
- Obtain Connected Components Workbench ... 9
- Use Connected Components Workbench ... 9

Controller Changes in Run Mode ... 9
Using Run Mode Change (RMC) ... 10
- **Uncommitted Changes ... 11**
- RMC Memory ... 12
- Limitations of RMC ... 14

Using Run Mode Configuration Change (RMCC) ... 15
- Using Modbus RTU Communication ... 16
- Using EtherNet/IP Communication ... 17

Agency Certifications ... 19
Compliance to European Union Directives ... 19
- EMC Directive ... 20
- Low Voltage Directive ... 20

Installation Considerations ... 20
- Environment and Enclosure ... 22
- Preventing Electrostatic Discharge ... 22

Safety Considerations ... 22
- North American Hazardous Location Approval ... 23
- Disconnecting Main Power ... 23
- Safety Circuits ... 24
- Power Distribution ... 24
- Periodic Tests of Master Control Relay Circuit ... 24

Power Considerations ... 25
- Isolation Transformers ... 25
- Power Supply Inrush ... 25
- Loss of Power Source ... 25
- Input States on Power Down ... 26
- Other Types of Line Conditions ... 26

Preventing Excessive Heat ... 26
Master Control Relay ... 27

Using Emergency-Stop Switches 28
Schematic – Using IEC Symbols 29
Schematic – Using ANSI/CSA Symbols) 30

Chapter 3

Install Your Controller

Controller Mounting Dimensions 31
Mounting Dimensions 31
DIN Rail Mounting. 33
Panel Mounting 34
Panel Mounting Dimensions 34
System Assembly 37

Chapter 4

Wire Your Controller

Wiring Requirements and Recommendation 39
Use Surge Suppressors 40
Recommended Surge Suppressors 42
Grounding the Controller. 43
Wiring Diagrams 43
Controller I/O Wiring. 46
Minimize Electrical Noise 47
Analog Channel Wiring Guidelines 47
Minimize Electrical Noise on Analog Channels 47
Grounding Your Analog Cable 48
Wiring Examples 48
Embedded Serial Port Wiring 49

Chapter 5

Communication Connections

Overview 51
Supported Communication Protocols. 51
Modbus RTU 53
CIP Serial Client/Server (RS-232 only). 53
ASCII 53
Modbus TCP Client/Server 53
CIP Symbolic Client/Server 54
CIP Client Messaging. 55
Sockets Client/Server TCP/UDP. 55
CIP Communications Pass-thru 56
Examples of Supported Architectures. 56
Use Modems with Micro800 Controllers. 57
Making a DF1 Point-to-Point Connection. 57
Construct Your Own Modem Cable 58
Configure Serial Port 58
Configure CIP Serial Driver 59
Configure Modbus RTU 61

Configure ASCII 62
Configure Ethernet Settings 64
Validate IP Address 66
Ethernet Host Name 66
Configure CIP Serial Driver 67
OPC Support Using RSLinx Enterprise 67

Chapter 6

Program Execution in Micro800

Overview of Program Execution 69
Execution Rules 70
Optional Module 70
Controller Load and Performance Considerations 71
Periodic Execution of Programs 71
Power Up and First Scan 72
Variable Retention 72
Memory Allocation 73
Guidelines and Limitations for Advanced Users 73

Chapter 7

Motion Control

PTO Motion Control 75
Use the Micro800 Motion Control Feature 76
Input and Output Signals 78
Motion Control Function Blocks 81
General Rules for the Motion Control Function Blocks 83
Motion Axis and Parameters 91
Motion Axis State Diagram 92
Axis States 93
Limits 94
Motion Stop 96
Motion Direction 97
Axis Elements and Data Types 98
Axis Error Scenarios 99
MC_Engine_Diag Data Type 100
Function Block and Axis Status Error Codes 100
Major Fault Handling 103
Motion Axis Configuration in Connected Components Workbench 103
Add New Axis 104
Edit Axis Configuration 105
Axis Start/Stop Velocity 112
Real Data Resolution 112
PTO Pulse Accuracy 114
Motion Axis Parameter Validation 115
Delete an Axis 115
Monitor an Axis 115
Homing Function Block 116

Conditions for Successful Homing 117
MC_HOME_ABS_SWITCH 117
MC_HOME_LIMIT_SWITCH 119
MC_HOME_REF_WITH_ABS. 120
MC_HOME_REF_PULSE 122
MC_HOME_DIRECT 123
Use PTO for PWM Control 124
POU PWM_Program. 125
HSC Feedback Axis 126

Chapter 8

Use the High-Speed Counter and Programmable Limit Switch

High-Speed Counter Overview 127
Programmable Limit Switch Overview 127
What is High-Speed Counter?. 128
Features and Operation 128
HSC Inputs and Wiring Mapping 129
High Speed Counter (HSC) Data Structures 133
HSC APP Data Structure 133
PLS Enable (HSCAPP.PLSEnable) 133
HSCID (HSCAPP.HSCID) 134
HSC Mode (HSCAPP.HSCMode) 134
Accumulator (HSCAPP. Accumulator) 140
High Preset (HSCAPP.HPSetting) 140
Low Preset (HSCAPP.LPSetting). 141
Overflow Setting (HSCAPP.OFSetting) 141
Underflow Setting (HSCAPP.UFSetting) 141
Output Mask Bits (HSCAPP.OutputMask) 142
High Preset Output (HSCAPP.HPOutput) 143
Low Preset Output (HSCAPP.LPOutput) 143
HSC STS (HSC Status) Data Structure 144
Counting Enabled (HSCSTS.CountEnable). 144
Error Detected (HSCSTS.ErrorDetected) 144
Count Up (HSCSTS.CountUpFlag). 145
Count Down (HSCSTS.CountDownFlag) 145
Mode Done (HSCSTS.Mode1Done) 145
Overflow (HSCSTS.OVF) 145
Underflow (HSCSTS.UNF) 146
Count Direction (HSCSTS.CountDir) 146
High Preset Reached (HSCSTS.HPReached) 146
Low Preset Reached (HSCSTS.LPReached) 147
Overflow Interrupt (HSCSTS.OFCauseInter) 147
Underflow Interrupt (HSCSTS.UFCauseInter). 147
High Preset Interrupt (HSCSTS.HPCauseInter). 148
Low Preset Interrupt (HSCSTS.LPCauseInter) 148
Programmable Limit Switch Position (HSCSTS.PLSPosition) . . 148

Error Code (HSCSTS.ErrorCode) 149
Accumulator (HSCSTS.Accumulator) 149
High Preset (HSCSTS.HP) 149
Low Preset (HSCSTS.LP) 150
High Preset Output (HSCSTS.HPOutput) 150
Low Preset Output (HSCSTS.LPOutput) 150
HSC (High Speed Counter) Function Block 151
HSC Commands (HScCmd) 151
HSC_SET_STS Function Block 153
Programmable Limit Switch (PLS) Function 153
PLS Data structure 154
PLS Operation 154
PLS Example 155
HSC Interrupts 156
HSC Interrupt Configuration 157
HSC Interrupt POU 158
Auto Start (HSC0.AS) 158
Mask for IV (HSC0.MV) 158
Mask for IN (HSC0.MN) 158
Mask for IH (HSC0.MH) 159
Mask for IL (HSC0.ML) 159
HSC Interrupt Status Information 159
User Interrupt Enable (HSC0.Enabled) 159
User Interrupt Executing (HSC0.EX) 159
User Interrupt Pending (HSC0.PE) 160
User Interrupt Lost (HSC0.LS) 160
Use HSC 160

Chapter 9

Controller Security

..... 161
Exclusive Access 161
Password Protection 161
Compatibility 162
Work with a Locked Controller 163
Upload from a Password-Protected Controller 163
Debug a Password-Protected Controller 163
Download to a Password-Protected Controller 163
Transfer Controller Program and Password-Protect Receiving Controller 164
Back Up and Restore a Password-Protected Controller 165
Configure Controller Password 165
Recover from a Lost Password 166

Appendix A

Specifications

Micro830 Controllers 167
Micro830 10-Point Controllers 167
Micro830 16-Point Controllers 171
Micro830 24-Point Controllers 174
Micro830 48-Point Controllers 178
Micro830 and Micro850 Relay Charts 183
Micro850 Controllers 183
Micro850 24-Point Controllers 184
Micro850 48-Point Controllers 187
Micro800 Programmable Controller External AC Power Supply 193

Appendix B

Modbus Mapping for Micro800

Modbus Mapping 195
Endian Configuration 195
Mapping Address Space and supported Data Types 195
Example 1, PanelView Component HMI (Master) to Micro800 (Slave) 196
Example 2, Micro800 (Master) to PowerFlex 4M Drive (Slave) .. 197
Performance 200

Appendix C

Quickstarts

Flash Upgrade Your Micro800 Firmware 201
Establish Communications Between RSLinx and a Micro830/Micro850 Controller through USB 204
Configure Controller Password 210
Set Controller Password 211
Change Password 212
Clear Password 213
Use the High Speed Counter 214
Create the HSC Project and Variables 216
Assign Values to the HSC Variables 219
Assign Variables to the Function Block 222
Run the High Speed Counter 223
Use the Programmable Limit Switch (PLS) Function 225
Forcing I/Os 227
Checking if Forces (locks) are Enabled 227
I/O Forces After a Power Cycle 228
Using Run Mode Change 229
Create the Project 229
Edit the Project Using Run Mode Change 232

Appendix D

User Interrupts

Information About Using Interrupts 237
What is an Interrupt? 237
When Can the Controller Operation be Interrupted? 238
Priority of User Interrupts 238
User Interrupt Configuration 240
User Fault Routine 240
User Interrupt Instructions 241
STIS - Selectable Timed Start 241
UID - User Interrupt Disable 242
UIE - User Interrupt Enable 244
UIF - User Interrupt Flush 245
UIC – User Interrupt Clear 246
Using the Selectable Timed Interrupt (STI) Function 247
Selectable Time Interrupt (STI) Function Configuration and Status 247
STI Function Configuration 248
STI Function Status Information 248
Using the Event Input Interrupt (EII) Function 249
Event Input Interrupt (EII) Function Configuration and Status 250
EII Function Configuration 250
EII Function Status Information 251

Appendix E

Troubleshooting

Status Indicators on the Controller 253
Normal Operation 254
Error codes 255
Fault Types 255
Corrective Action for Recoverable and Non-recoverable Faults .. 263
Controller Error Recovery Model 264
Calling Rockwell Automation for Assistance 265

Appendix F

PID Function Blocks

..... 267
PID Function Block 268
IPIDCONTROLLER Function Block 271
How to Autotune 274
How Autotune Works 275
Troubleshooting an Autotune Process 275
PID Application Example 276
PID Code Sample 277

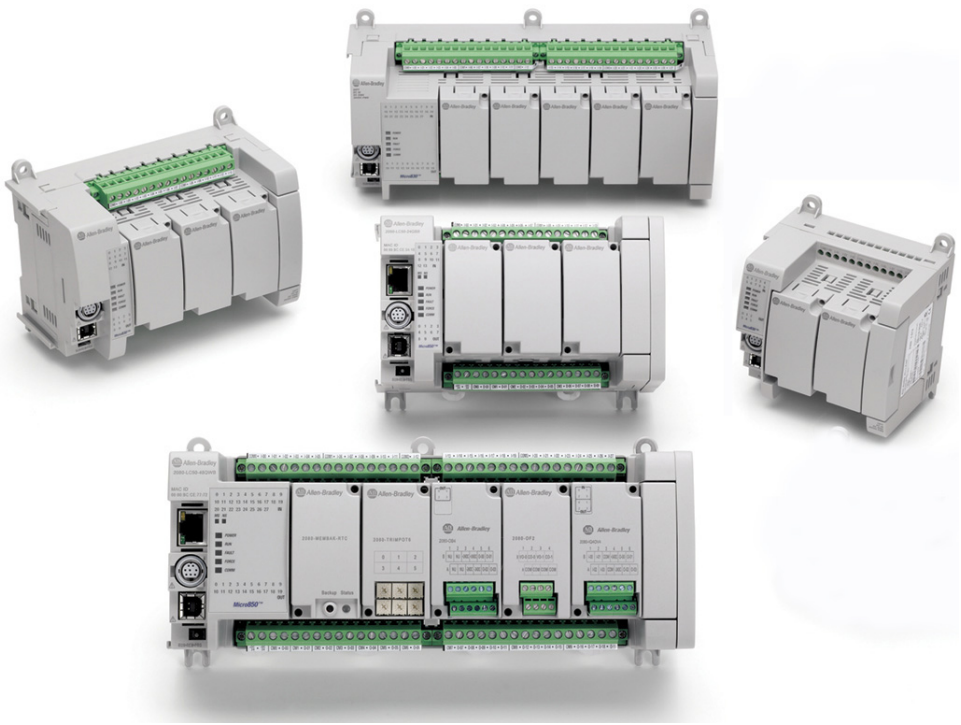
Appendix G

System Loading

Calculate Total Power for Your Micro830/Micro850 Controller 279
..... 280

Index 281

Hardware Overview



This chapter provides an overview of the Micro830 and Micro850 hardware features. It has the following topics:

Topic	Page
Hardware Features	1
Micro830 Controllers	2
Micro850 Controllers	4
Programming Cables	6
Embedded Serial Port Cables	7
Embedded Ethernet Support	7

Hardware Features

Micro830 and Micro850 controllers are economical brick style controllers with embedded inputs and outputs. Depending on the controller type, it can accommodate from two to five plug-in modules. The Micro850 controller has expandable features and can additionally support up to four expansion I/O modules.

IMPORTANT For information on supported plug-in modules and expansion I/O, see the following publications:

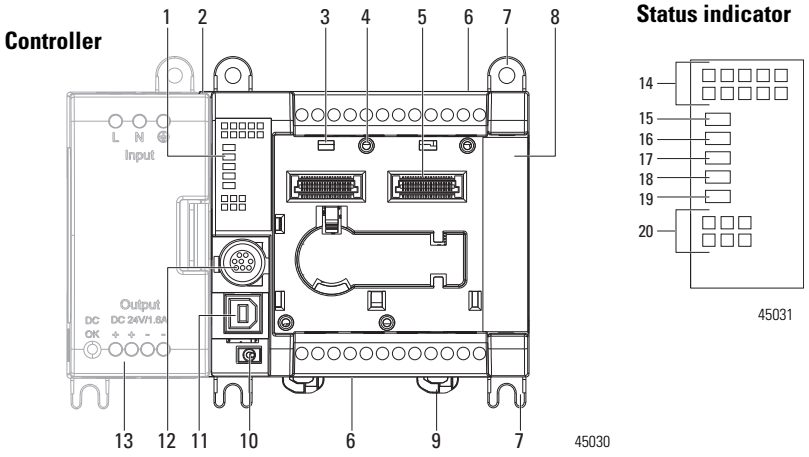
- Micro800 Discrete and Analog Expansion I/O User Manual, publication 2080-UM003
- Micro800 Plug-in Modules User Manual, publication 2080-UM004

The controllers also accommodate any class 2 rated 24V DC output power supply that meets minimum specifications such as the optional Micro800 power supply.

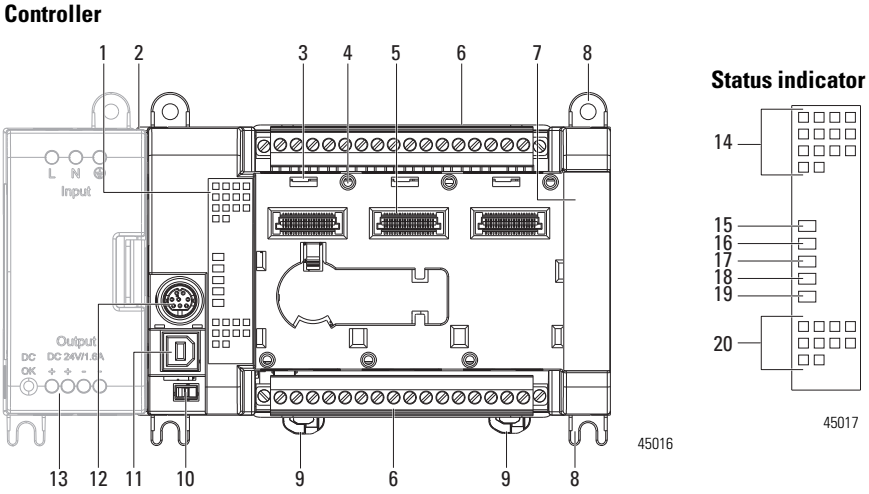
See Troubleshooting on page 253 for descriptions of status indicator operation for troubleshooting purposes.

Micro830 Controllers

Micro830 10/16-point controllers and status indicators

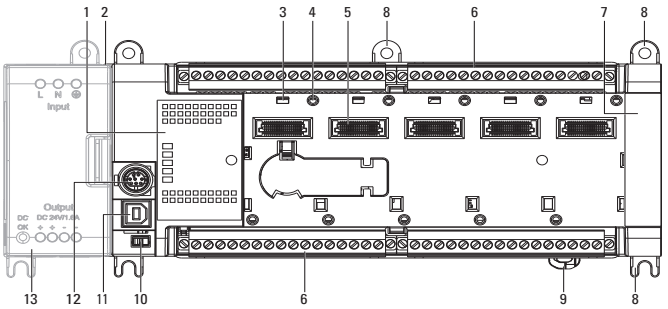


Micro830 24-point controllers and status indicators

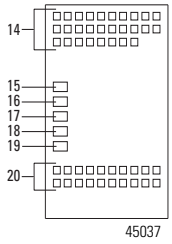


Micro830 48-point controllers and status indicators

Controller



Status indicator



Controller Description

	Description		Description
1	Status indicators	8	Mounting screw hole / mounting foot
2	Optional power supply slot	9	DIN rail mounting latch
3	Plug-in latch	10	Mode switch
4	Plug-in screw hole	11	Type B connector USB port
5	40 pin high speed plug-in connector	12	RS-232/RS-485 non-isolated combo serial port
6	Removable I/O terminal block	13	Optional AC power supply
7	Right-side cover		

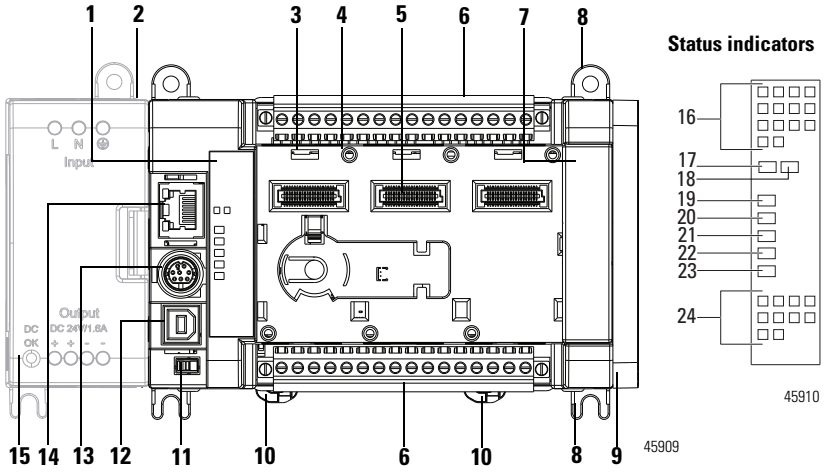
Status Indicator Description(1)

	Description		Description
14	Input status	18	Force status
15	Power status	19	Serial communications status
16	Run status	20	Output status
17	Fault status		

(1) For detailed description of the different status LED indicators, see Troubleshooting on page 253.

Micro850 Controllers

Micro850 24-point controllers and status indicators



Controller Description

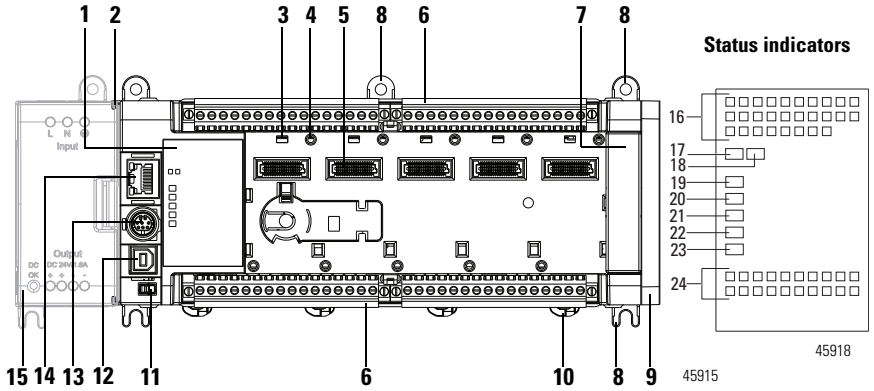
	Description		Description
1	Status indicators	9	Expansion I/O slot cover
2	Optional power supply slot	10	DIN rail mounting latch
3	Plug-in latch	11	Mode switch
4	Plug-in screw hole	12	Type B connector USB port
5	40 pin high speed plug-in connector	13	RS232/RS485 non-isolated combo serial port
6	Removable I/O terminal block	14	RJ-45 Ethernet connector (with embedded green and yellow LED indicators)
7	Right-side cover	15	Optional power supply
8	Mounting screw hole / mounting foot		

Status Indicator Description(1)

	Description		Description
16	Input status	21	Fault status
17	Module Status	22	Force status
18	Network Status	23	Serial communications status
19	Power status	24	Output status
20	Run status		

(1) For detailed descriptions of the different status LED indicators, see Troubleshooting on page 253.

Micro850 48-point controllers and status indicators



Controller Description

	Description		Description
1	Status indicators	9	Expansion I/O slot cover
2	Optional power supply slot	10	DIN rail mounting latch
3	Plug-in latch	11	Mode switch
4	Plug-in screw hole	12	Type B connector USB port
5	40-pin high speed plug-in connector	13	RS232/RS485 non-isolated combo serial port
6	Removable I/O terminal block	14	RJ-45 EtherNet/IP connector (with embedded yellow and green LED indicators)
7	Right-side cover	15	Optional AC power supply
8	Mounting screw hole / mounting foot		

Status Indicator Description(1)

	Description		Description
16	Input status	21	Fault status
17	Module status	22	Force status
18	Network status	23	Serial communications status
19	Power status	24	Output status
20	Run status		

(1) For detailed descriptions of these LED status indicators, see Troubleshooting on page 253.

Micro830 Controllers – Number and Type of Inputs/Outputs

Catalog Number	Inputs		Outputs			PTO Support	HSC Support
	110V AC	24V DC/V AC	Relay	24V Sink	24V Source		
2080-LC30-10QWB		6	4				2
2080-LC30-10QVB		6		4		1	2
2080-LC30-16AWB	10		6				
2080-LC30-16QWB		10	6				2

Micro830 Controllers – Number and Type of Inputs/Outputs

Catalog Number	Inputs		Outputs			PTO Support	HSC Support
	110V AC	24V DC/V AC	Relay	24V Sink	24V Source		
2080-LC30-16QVB		10		6		1	2
2080-LC30-24QBB		14			10	2	4
2080-LC30-24QVB		14		10		2	4
2080-LC30-24QWB		14	10				4
2080-LC30-48AWB	28		20				
2080-LC30-48QBB		28			20	3	6
2080-LC30-48QVB		28		20		3	6
2080-LC30-48QWB		28	20				6

Micro850 Controllers – Number and Types of Inputs and Outputs

Catalog Number	Inputs		Outputs			PTO Support	HSC Support
	120V AC	24V DC/V AC	Relay	24V Sink	24V Source		
2080-LC50-24AWB	14		10				
2080-LC50-24QBB		14			10	2	4
2080-LC50-24QVB		14		10		2	4
2080-LC50-24QWB		14	10				4
2080-LC50-48AWB	28		20				
2080-LC50-48QBB		28			20	3	6
2080-LC50-48QVB		28		20		3	6
2080-LC50-48QWB		28	20				6

Programming Cables

Micro800 controllers have a USB interface, making standard USB cables usable as programming cables.

Use a standard USB A Male to B Male cable for programming the controller.

45221

Embedded Serial Port Cables

Embedded serial port cables for communication are listed here. All embedded serial port cables must be 3 meters in length, or shorter.

Embedded Serial Port Cable Selection Chart

Connectors	Length	Cat. No.		Connectors	Length	Cat. No.
8-pin Mini DIN to 8-pin Mini DIN	0.5 m (1.5 ft)	1761-CBL-AM00(1)		8-pin Mini DIN to 9-pin D Shell	0.5 m (1.5 ft)	1761-CBL-AP00(1)
8-pin Mini DIN to 8-pin Mini DIN	2 m (6.5 ft)	1761-CBL-HM02(1)		8-pin Mini DIN to 9-pin D Shell	2 m (6.5 ft)	1761-CBL-PM02(1)
				8-pin Mini DIN to 6-pin RS-485 terminal block	30 cm (11.8in.)	1763-NC01 series A

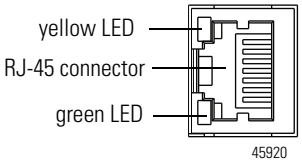
(1) Series C or later for Class 1 Div 2 applications.

Embedded Ethernet Support

For Micro850 controllers, a 10/100 Base-T Port (with embedded green and yellow LED indicators) is available for connection to an Ethernet network through any standard RJ-45 Ethernet cable. The LED indicators serve as indicators for transmit and receive status.

RJ-45 Ethernet Port Pin Mapping

Contact Number	Signal	Direction	Primary Function
1	TX+	OUT	Transmit data +
2	TX-	OUT	Transmit data -
3	RX+	IN	Differential Ethernet Receive Data +
4			Terminated
5			Terminated
6	RX-	IN	Differential Ethernet Receive Data -
7			Terminated
8			Terminated
Shield			Chassis Ground



The yellow status LED indicates Link (solid yellow) or No Link (off).

The green status LED indicates activity (blinking green) or no activity (off).

Micro850 controllers support Ethernet crossover cables (2711P-CBL-EX04).

Ethernet Status Indication

Micro850 controllers also support two LEDs for EtherNet/IP to indicate the following:

- Module status
- Network status

See Troubleshooting on page 253 for descriptions of Module and Network status indicators.

Notes:

About Your Controller

Programming Software for Micro800 Controllers

Connected Components Workbench is a set of collaborative tools supporting Micro800 controllers. It is based on Rockwell Automation and Microsoft Visual Studio technology and offers controller programming, device configuration and integration with HMI editor. Use this software to program your controllers, configure your devices and design your operator interface applications.

Connected Components Workbench provides a choice of IEC 61131-3 programming languages (ladder diagram, function block diagram, structured text) with user defined function block support that optimizes machine control.

Obtain Connected Components Workbench

A free download is available at:

http://www.rockwellautomation.com/global/products-technologies/connected-components/.

Use Connected Components Workbench

To help you program your controller through the Connected Components Workbench software, you can refer to the Connected Components Workbench Online Help (it comes with the software).

Controller Changes in Run Mode

Micro820/Micro830/Micro850 controllers allow you to make certain changes while in run mode by using the following features:

- Run Mode Change (RMC)
  This feature allows logic modifications to a running project without going to remote program mode.
  For more information, see Using Run Mode Change (RMC) on page 10.
- Run Mode Configuration Change (RMCC)|
  This feature allows changing the address configuration of the controller to| be made within a program during run mode.
  For more information, see Using Run Mode Configuration Change (RMCC) on page 15.

Using Run Mode Change (RMC)

Run Mode Change (RMC) is a productivity enhancement feature introduced in Release 8 for Micro820/Micro830/Micro850 controllers. It saves the user time by allowing logic modifications to a running project without going to remote program mode and without disconnecting from the controller. It requires the Connected Components Workbench Developer Edition Release 8 software to use this feature.

IMPORTANT Micro820/Micro830/Micro850 controller firmware revision 8.xxx or higher is also required to use Run Mode Change.

RMC is useful when the user is developing a project by incrementally adding small changes to the logic and immediately wants to see the effects of the changes on the machine. With RMC, since the controller stays in remote run mode, the controller logic and machine actuators will not have to constantly reinitialize, which can occur if the controller is switched to remote program mode (for example, first scan bit is checked in program logic to clear outputs).

When user is editing, building, and downloading a project without using RMC, a full build of the entire controller project is performed and also a full download of the project is performed. During RMC an incremental build is performed and only incremental changes are downloaded to the controller.

IMPORTANT Do not disconnect from the controller after performing Run Mode Change, do a full build, and try to reconnect. Connected Components Workbench will treat the project in the controller as different from the project in Connected Components Workbench, and ask to either upload or download even though the logic is identical.

RMC is performed incrementally at the end of every program scan in order to prevent a large delay in the program scan. This adds up to an additional 12 ms to the scan time. For example, if the program scan is normally 10 ms, it may increase to 22 ms during RMC until the update is finished. Similarly user interrupts may be delayed.

Example of the Benefits of Using RMC – 20% Reduction in Download Time

Number of Changes	Time to Perform Conventional Download (seconds)	Time to Test Logic and Accept Changes (seconds)
1	36	29
5	180	130
10	360	255

Memory size of project used for comparison:
Data = 14784 bytes; Program = 2352 bytes

Note: The duration starts when the RMC button is clicked while connected to the controller and ends when the accept is finished. For example:
1. When connected to the controller, click RMC
2. Modify program
3. Click Test Logic
4. Click Accept to finish, or click Test Logic to make another change

ATTENTION: Use extreme caution when you use Run Mode Change. Mistakes can injure personnel and damage equipment. Before using Run Mode Change:

- assess how machinery will respond to the changes.
- notify all personnel about the changes.

A new global variable __SYSVA_PROJ_INCOMPLETE has been added to indicate when Run Mode Changes are being made. This can be used to notify personnel on the HMI that there are uncommitted changes in the controller.

Bit Definitions of Global Variable – __SYSVA_PROJ_INCOMPLETE

Bit	Definition
0	Set when the Run Mode Change process starts. Cleared once the Run Mode Change is written permanently to the controller (completion of Accept or Undo). This bit can be used to warn operators that a run mode change is in progress and that there are uncommitted changes in the controller.
1	Set if an error occurred while saving the changes to flash or an integrity check failed during Run Mode Change. Cleared on the next successful Run Mode change.

When you perform a Test Logic Change, the value of the variable is changed from zero to one. After you choose to accept or undo the changes, the value of the variable is reset to zero.

IMPORTANT When a Test Logic is performed, or undoing changes after the Test Logic is completed, any active communication instructions will be aborted while the changes are downloaded to the controller.

Uncommitted Changes

Uncommitted changes are changes made in RMC that have not been accepted or undone after a Test Logic Change has been performed.

If the controller power loses power while there are uncommitted changes, you will not be able to re-enter RMC upon reconnection. You can choose to re-download the project to keep the changes, or upload if the uncommitted changes are not wanted.

If you choose to upload a project with uncommitted changes from the controller, you cannot enter RMC until you have done a full download.

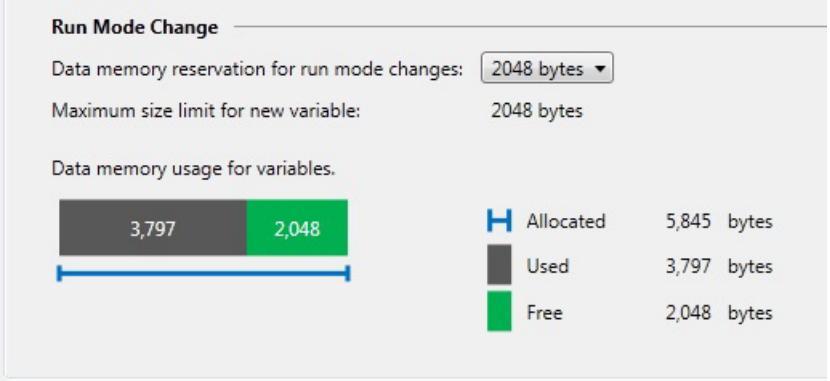
RMC Memory

Run Mode Change (RMC) memory is used to store both the logic and user variable changes made during RMC. The default amount of memory allocated is 2KB and can be increased up to 8KB. However there is still a limit of 2KB for logic and user variables changes per Test Logic. To adjust the amount of RMC memory, the controller must be offline. After you have adjusted the amount, you must build the project and download it to the controller.

IMPORTANT In a Connected Components Workbench Release 8 project, the available user data space was reduced by 6 KB to support optimal project settings for the new RMC feature.

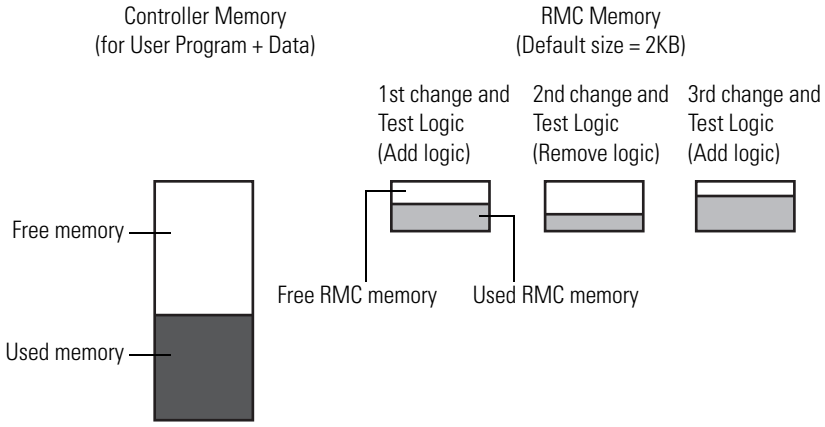
If you have a project that was developed before Release 8, you may need to reduce the default "Allocated" 8 KB Temporary Variables section from the Memory page in order to compile the project successfully.

Controller Memory Diagnostics Page in Connected Components Workbench



During RMC an incremental build is performed and only incremental changes are downloaded to the controller until the RMC memory has been filled.

RMC Memory Usage Example

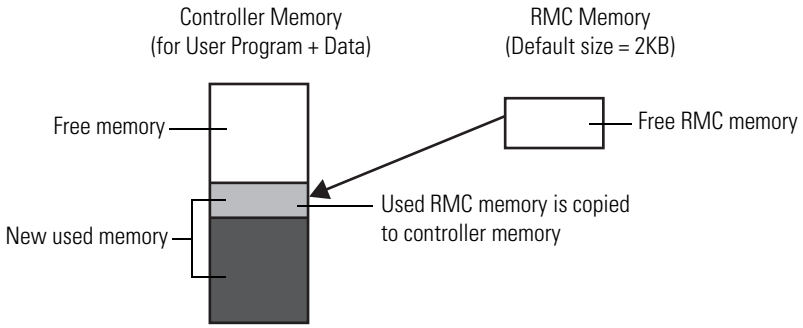


If not enough RMC memory is available to make more changes (for example, a "not enough memory" error message appears during RMC build or Test Logic), then a full download must be performed to transfer the incremental changes from the RMC memory to standard user program and data memory.

Transferring Contents in RMC Memory to Controller Memory

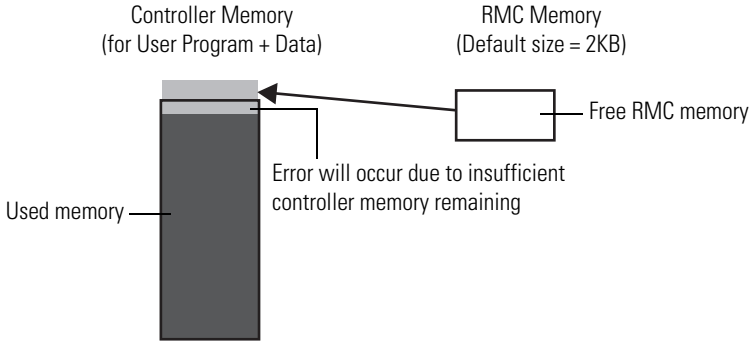
The changes that you have made during RMC are stored in RMC memory and will remain there until you perform a full build and download (while the controller is disconnected).

RMC Memory Usage When Performing Full Build and Download Example



However if the controller memory does not have enough space remaining to copy the contents of the RMC memory as shown below, the operation will fail and a "not enough memory" error message will appear. Do not use RMC if you are near the limits of your controller memory.

Insufficient Controller Memory Example



Limitations of RMC

Take note of the following limitations when using the Run Mode Change (RMC) feature:

- Configuration changes cannot be made (for example, change filter times).
- Up to 2KB of logic (approximately 150 boolean instructions) and user variables and can be added for each Test Logic.
- Total memory allocated for RMC (cumulative of all Test Logic Changes) can be increased from 2KB to 8KB, but the 2KB limit for logic and user variables per Test Logic remains.
- Up to 20 POU (Program Organizational Units) can be added for each change (for example, if you currently have 5 POU, you can add 20 more for a total of 25 POU).
- If a User Defined Function Block is modified that changes the local variables, the local variables will be reinitialized or reset to zero and a warning message will be shown during the build. If you want to reapply the initial value, right-click on the UDFB and select Refactor -> Reset Initial Values of Instances.
- RMC is not possible after doing a Discover Project operation if a new module is detected because the configuration has changed.
- Exchange files cannot be imported when in RMC because it is considered a configuration change.
- Making changes to the display configuration (for example, hiding comments) are treated as logic changes and require you to build the project.
- Global variables cannot be deleted or modified in RMC, but can be added. To delete or modify a global variable, Connected Components Workbench must be disconnected from the controller.
- When using CIP messaging in RMC, setting the CIPTARGETCFG data type parameter ConnClose to TRUE has no effect. The Ethernet session does not close immediately upon successful messaging and you have to wait for the connection to timeout after 60 seconds. This applies to Release 9 or earlier projects. For Release 10 or later projects, the CIP connection timeout is configurable.

WARNING: If you delete the output rung when in Run Mode Change and accept the changes, the output on the controller will remain ON.

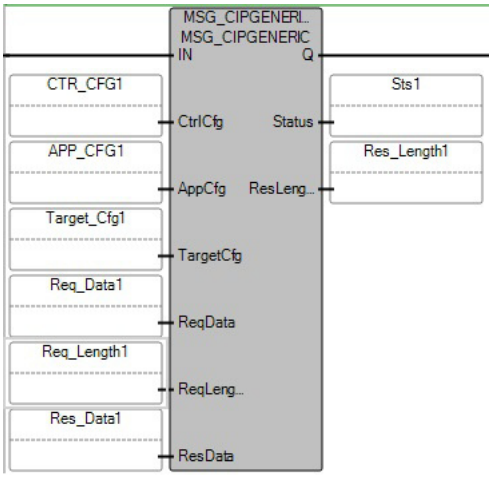
See Using Run Mode Change on page 229 for an example on how to use this feature.

Using Run Mode Configuration Change (RMCC)

Run Mode Configuration Change (RMCC) is a productivity enhancement feature introduced in Release 9 for Micro820/Micro830/Micro850 controllers. It allows users to reuse an identical program with multiple controllers simply by changing the address configuration of a controller within the program during run mode. Micro820/Micro830/Micro850 controller firmware revision 9.xxx or higher is required to use this feature.

RMCC can be used to change the address configuration of the controller during run mode when the communication protocol is set to Modbus RTU for serial ports or EtherNet/IP for the Ethernet port. RMCC uses a CIP Generic message which can only be sent from within a controller program and not from an external device to the controller.

CIP Generic Message Instruction for Run Mode Configuration Change



Run Mode Configuration Change (RMCC) can only be performed by the controller that is sending the message. To do that, you need to configure the CIP Generic message as a loop-back message by setting the path to “0,0”.

Configure CIP Generic Message as a Loop-back Message

Name	Data Type	Dimension	String Size	Initia
	CIPTARGETCFG			
Target_Cfg1	CIPTARGETCFG			...
Target_Cfg1.Path	STRING		80	'0,0'
Target_Cfg1.CipConnMode	USINT			0
Target_Cfg1.UcmmTimeout	UDINT			0
Target_Cfg1.ConnMsgTimeout	UDINT			0
Target_Cfg1.ConnClose	BOOL			

For Micro830/Micro850 controllers, the address configuration change is permanent and will retained when the controller is power cycled.

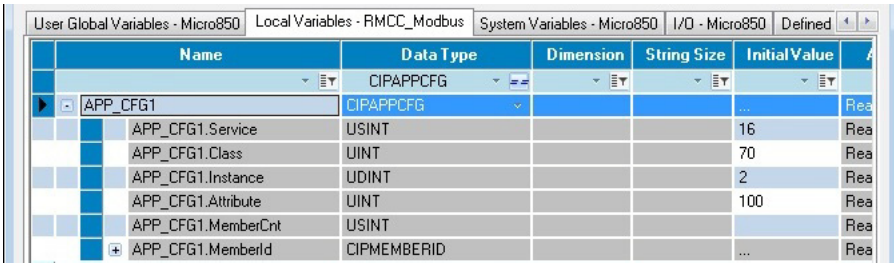
Using Modbus RTU Communication

To use RMCC with the Modbus RTU communication protocol, the serial port must be set to the Modbus slave role. A CIP Generic message is sent from within a program with the following parameters.

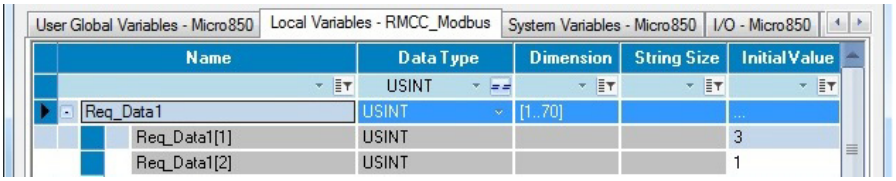
CIP Generic Message Parameters for RMCC using Modbus RTU

Parameter	Value
Service	16
Class	70
Instance	2 – Embedded serial port 5, 6, 7, 8, or 9 – Plug-in modules
Attribute	100
ReqData	New node address, 1
ReqLen	2

RMCC Modbus Example – Set the Parameters

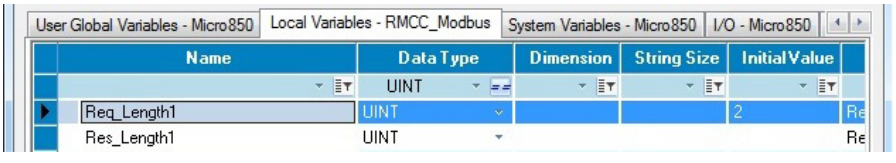


RMCC Modbus Example – Set the New Node Address



The first byte indicates the new node address for the controller. For this example, the new node address is "3". The second byte must always be "1", this indicates that the Modbus role is configured as Slave.

RMCC Modbus Example – Set the Message Length

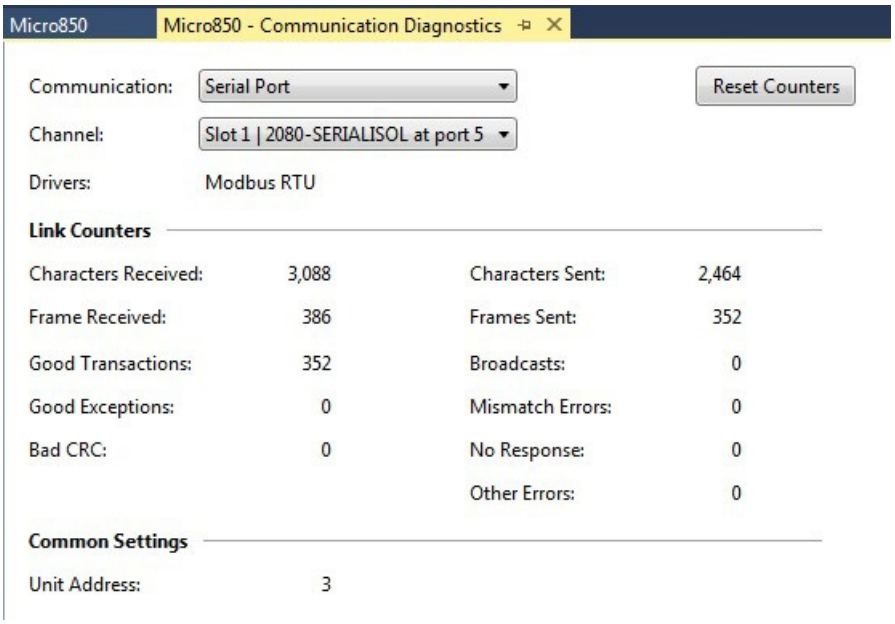


When the new node address is configured and applied, the port is not restarted.

IMPORTANT You must ensure that the new node address being configured is unique as it will not be checked against existing node addresses of other devices.

You can verify that the node address has changed after performing RMCC by looking at the Communication Diagnostics tab for the controller.

RMCC Modbus Example – Verify Address Change



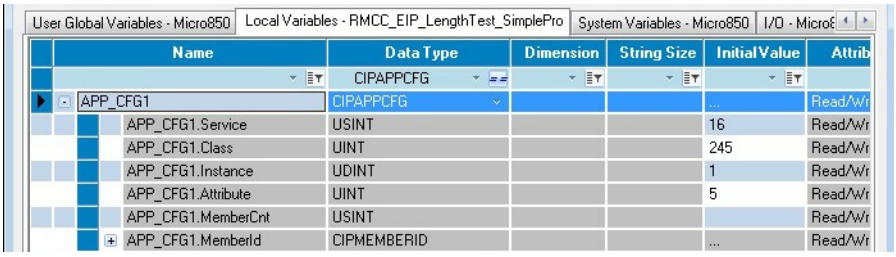
Using EtherNet/IP Communication

To use RMCC with the EtherNet/IP communication protocol, the controller must be configured to use a static IP address. If the controller is configured to use BOOTP or DHCP, the change will be rejected. A CIP Generic message is sent from within a program with the following parameters.

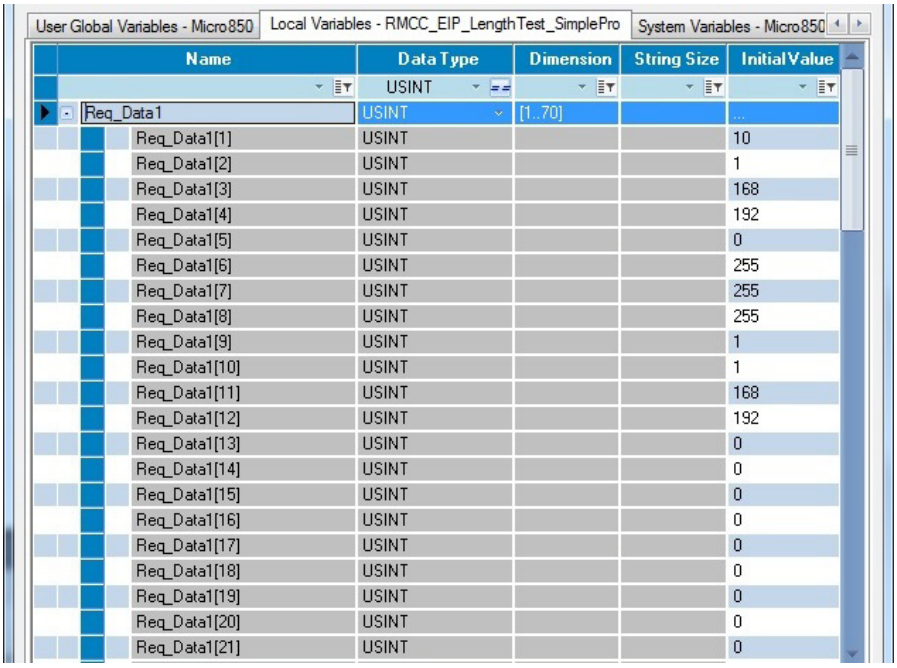
CIP Generic Message Parameters for RMCC using EtherNet/IP

Parameter	Value
Service	16
Class	245
Instance	1
Attribute	5
ReqData	IP address, Subnet mask, Gateway address
ReqLen	22 bytes

RMCC EtherNet/IP Example – Set the Parameters



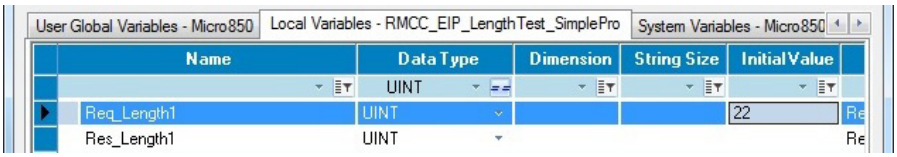
RMCC EtherNet/IP Example – Set the New IP Address



For this example, the new IP Address is set to the following:

- IP address = 192.168.1.10
- Subnet mask = 255.255.255.0
- Gateway address = 192.168.1.1

RMCC EtherNet/IP Example – Set the Message Length



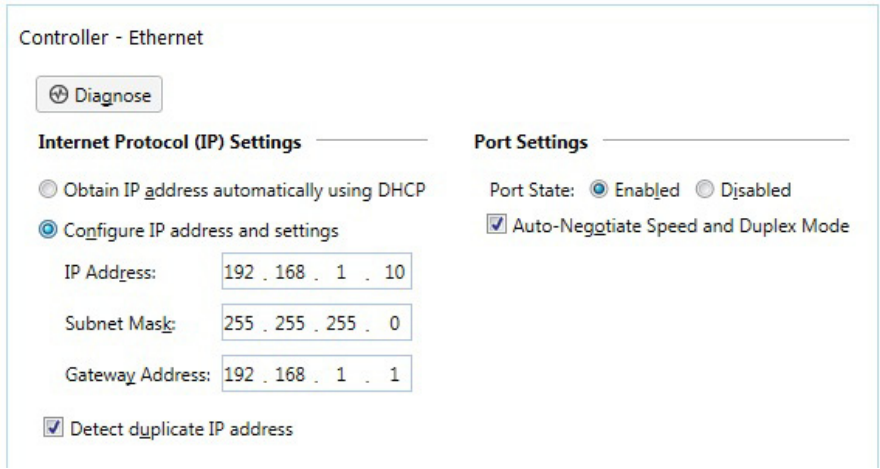
After the new IP address is configured and applied, the controller will disconnect from Connected Components Workbench if communication is through Ethernet.

IMPORTANT Micro830 controllers do not support Run Mode Configuration Change using EtherNet/IP.

IMPORTANT You should not perform IP address changes continuously. Allow an interval of at least six seconds before performing the next IP address change in order for duplicate address detection to work properly.

You can verify that the IP address has changed after performing RMCC by looking at the Ethernet settings for the controller.

RMCC EtherNet/IP Example – Verify Address Change


Controller - Ethernet
Diagnose
Internet Protocol (IP) Settings
Obtain IP address automatically using DHCP
Configure IP address and settings
IP Address: 192 . 168 . 1 . 10
Subnet Mask: 255 . 255 . 255 . 0
Gateway Address: 192 . 168 . 1 . 1
Detect duplicate IP address
Port Settings
Port State: Enabled Disabled
Auto-Negotiate Speed and Duplex Mode

Agency Certifications

- UL Listed Industrial Control Equipment, certified for US and Canada. UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada.
- CE marked for all applicable directives
- C-Tick marked for all applicable acts
- KC - Korean Registration of Broadcasting and Communications Equipment, compliant with: Article 58-2 of Radio Waves Act, Clause 3.

Compliance to European Union Directives

This product has the CE mark and is approved for installation within the European Union and EEA regions. It has been designed and tested to meet the following directives.

EMC Directive

This product is tested to meet Council Directive 2004/108/EC Electromagnetic Compatibility (EMC) and the following standards, in whole or in part, documented in a technical construction file:

- EN 61131-2; Programmable Controllers (Clause 8, Zone A & B)
- EN 61131-2; Programmable Controllers (Clause 11)
- EN 61000-6-4
  EMC - Part 6-4: Generic Standards - Emission Standard for Industrial Environments
- EN 61000-6-2
  EMC - Part 6-2: Generic Standards - Immunity for Industrial Environments

This product is intended for use in an industrial environment.

Low Voltage Directive

This product is tested to meet Council Directive 2006/95/ECLow Voltage, by applying the safety requirements of EN 61131-2 Programmable Controllers, Part 2 - Equipment Requirements and Tests.

For specific information required by EN 61131-2, see the appropriate sections in this publication, as well as the following Allen-Bradley publications:

- *Industrial Automation Wiring and Grounding Guidelines for Noise Immunity*, publication 1770-4.1.
- *Guidelines for Handling Lithium Batteries*, publication AG-5.4
- *Automation Systems Catalog*, publication B115

Installation Considerations

Most applications require installation in an industrial enclosure (Pollution Degree 2(1)) to reduce the effects of electrical interference (Over Voltage Category II(2)) and environmental exposure.

Locate your controller as far as possible from power lines, load lines, and other sources of electrical noise such as hard-contact switches, relays, and AC motor drives. For more information on proper grounding guidelines, see the *Industrial Automation Wiring and Grounding Guidelines* publication 1770-4.1.

(1) Pollution Degree 2 is an environment where normally only non-conductive pollution occurs except that occasionally temporary conductivity caused by condensation shall be expected.

(2) Overvoltage Category II is the load level section of the electrical distribution system. At this level, transient voltages are controlled and do not exceed the impulse voltage capability of the products insulation.

WARNING: When used in a Class I, Division 2, hazardous location, this equipment must be mounted in a suitable enclosure with proper wiring method that complies with the governing electrical codes.

WARNING: If you connect or disconnect the serial cable with power applied to this module or the serial device on the other end of the cable, an electrical arc can occur. This could cause an explosion in hazardous location installations. Be sure that power is removed or the area is nonhazardous before proceeding.

WARNING: The local programming terminal port is intended for temporary use only and must not be connected or disconnected unless the area is assured to be nonhazardous.

WARNING: The USB port is intended for temporary local programming purposes only and not intended for permanent connection.If you connect or disconnect the USB cable with power applied to this module or any device on the USB network, an electrical arc can occur. This could cause an explosion in hazardous location installations. Be sure that power is removed or the area is nonhazardous before proceeding.
The USB port is a nonincendive field wiring connection for Class I, Division2 Groups A, B, C and D.

WARNING: Exposure to some chemicals may degrade the sealing properties of materials used in the Relays. It is recommended that the User periodically inspect these devices for any degradation of properties and replace the module if degradation is found.

WARNING: If you insert or remove the plug-in module while backplane power is on, an electrical arc can occur. This could cause an explosion in hazardous location installations. Be sure that power is removed or the area is nonhazardous before proceeding.

WARNING: When you connect or disconnect the Removable Terminal Block (RTB) with field side power applied, an electrical arc can occur. This could cause an explosion in hazardous location installations.

WARNING: Be sure that power is removed or the area is nonhazardous before proceeding.

ATTENTION: To comply with the CE Low Voltage Directive (LVD), this equipment must be powered from a source compliant with the following: Safety Extra Low Voltage (SELV) or Protected Extra Low Voltage (PELV).

ATTENTION: To comply with UL restrictions, this equipment must be powered from a Class 2 source.

ATTENTION: Be careful when stripping wires. Wire fragments that fall into the controller could cause damage. Once wiring is complete, make sure the controller is free of all metal fragments.

ATTENTION: Do not remove the protective debris strips until after the controller and all other equipment in the panel near the module are mounted and wired. Remove strips before operating the controller. Failure to remove strips before operating can cause overheating.

ATTENTION: Electrostatic discharge can damage semiconductor devices inside the module. Do not touch the connector pins or other sensitive areas.

ATTENTION: The USB and serial cables are not to exceed 3.0 m (9.84 ft).

ATTENTION: Do not wire more than 2 conductors on any single terminal.

ATTENTION: Do not remove the Removable Terminal Block (RTB) until power is removed.

Environment and Enclosure

This equipment is intended for use in a Pollution Degree 2 industrial environment, in overvoltage Category II applications (as defined in IEC 60664-1), at altitudes up to 2000 m (6562 ft) without derating.

This equipment is considered Group 1, Class A industrial equipment according to IEC/CISPR 11. Without appropriate precautions, there may be difficulties with electromagnetic compatibility in residential and other environments due to conducted and radiated disturbances.

This equipment is supplied as open-type equipment. It must be mounted within an enclosure that is suitably designed for those specific environmental conditions that will be present and appropriately designed to prevent personal injury resulting from accessibility to live parts. The enclosure must have suitable flame-retardant properties to prevent or minimize the spread of flame, complying with a flame spread rating of 5VA, V2, V1, V0 (or equivalent) if non-metallic. The interior of the enclosure must be accessible only by the use of a tool. Subsequent sections of this publication may contain additional information regarding specific enclosure type ratings that are required to comply with certain product safety certifications.

In addition to this publication, see:

- Industrial Automation Wiring and Grounding Guidelines, Rockwell Automation publication 1770-4.1, for additional installation requirements.
- NEMA Standard 250 and IEC 60529, as applicable, for explanations of the degrees of protection provided by different types of enclosure.

Preventing Electrostatic Discharge

This equipment is sensitive to electrostatic discharge, which can cause internal damage and affect normal operation. Follow these guidelines when you handle this equipment:

- Touch a grounded object to discharge potential static.
- Wear an approved grounding wriststrap.
- Do not touch connectors or pins on component boards.
- Do not touch circuit components inside the equipment.
- Use a static-safe workstation, if available.
- Store the equipment in appropriate static-safe packaging when not in use.

Safety Considerations

Safety considerations are an important element of proper system installation. Actively thinking about the safety of yourself and others, as well as the condition of your equipment, is of primary importance. We recommend reviewing the following safety considerations.

North American Hazardous Location Approval

The following information applies when operating this equipment in hazardous locations:	**Informations sur l'utilisation de cet équipement en environnements dangereux:**
Products marked "CL I, DIV 2, GP A, B, C, D" are suitable for use in Class I Division 2 Groups A, B, C, D, Hazardous Locations and nonhazardous locations only. Each product is supplied with markings on the rating nameplate indicating the hazardous location temperature code. When combining products within a system, the most adverse temperature code (lowest "T" number) may be used to help determine the overall temperature code of the system. Combinations of equipment in your system are subject to investigation by the local Authority Having Jurisdiction at the time of installation.	Les produits marqués "CL I, DIV 2, GP A, B, C, D" ne conviennent qu'à une utilisation en environnements de Classe I Division 2 Groupes A, B, C, D dangereux et non dangereux. Chaque produit est livré avec des marquages sur sa plaque d'identification qui indiquent le code de température pour les environnements dangereux. Lorsque plusieurs produits sont combinés dans un système, le code de température le plus défavorable (code de température le plus faible) peut être utilisé pour déterminer le code de température global du système. Les combinaisons d'équipements dans le système sont sujettes à inspection par les autorités locales qualifiées au moment de l'installation.
EXPLOSION HAZARD • Do not disconnect equipment unless power has been removed or the area is known to be nonhazardous. • Do not disconnect connections to this equipment unless power has been removed or the area is known to be nonhazardous. Secure any external connections that mate to this equipment by using screws, sliding latches, threaded connectors, or other means provided with this product. • Substitution of any component may impair suitability for Class I, Division 2. • If this product contains batteries, they must only be changed in an area known to be nonhazardous.	**RISQUE D'EXPLOSION** • Couper le courant ou s'assurer que l'environnement est classé non dangereux avant de débrancher l'équipement. • Couper le courant ou s'assurer que l'environnement est classé non dangereux avant de débrancher les connecteurs. Fixer tous les connecteurs externes reliés à cet équipement à l'aide de vis, loquets coulissants, connecteurs filetés ou autres moyens fournis avec ce produit. • La substitution de tout composant peut rendre cet équipement inadapté à une utilisation en environnement de Classe I, Division 2. • S'assurer que l'environnement est classé non dangereux avant de changer les piles.

Disconnecting Main Power

WARNING: Explosion Hazard
Do not replace components, connect equipment, or disconnect equipment unless power has been switched off.

The main power disconnect switch should be located where operators and maintenance personnel have quick and easy access to it. In addition to disconnecting electrical power, all other sources of power (pneumatic and hydraulic) should be de-energized before working on a machine or process controlled by a controller.

Safety Circuits

WARNING: Explosion Hazard
Do not connect or disconnect connectors while circuit is live.

Circuits installed on the machine for safety reasons, like overtravel limit switches, stop push buttons, and interlocks, should always be hard-wired directly to the master control relay. These devices must be wired in series so that when any one device opens, the master control relay is de-energized, thereby removing power to the machine. Never alter these circuits to defeat their function. Serious injury or machine damage could result.

Power Distribution

There are some points about power distribution that you should know:

- The master control relay must be able to inhibit all machine motion by removing power to the machine I/O devices when the relay is de-energized. It is recommended that the controller remain powered even when the master control relay is de-energized.
- If you are using a DC power supply, interrupt the load side rather than the AC line power. This avoids the additional delay of power supply turn-off. The DC power supply should be powered directly from the fused secondary of the transformer. Power to the DC input and output circuits should be connected through a set of master control relay contacts.

Periodic Tests of Master Control Relay Circuit

Any part can fail, including the switches in a master control relay circuit. The failure of one of these switches would most likely cause an open circuit, which would be a safe power-off failure. However, if one of these switches shorts out, it no longer provides any safety protection. These switches should be tested periodically to assure they will stop machine motion when needed.

Power Considerations

The following explains power considerations for the micro controllers.

Isolation Transformers

You may want to use an isolation transformer in the AC line to the controller. This type of transformer provides isolation from your power distribution system to reduce the electrical noise that enters the controller and is often used as a step-down transformer to reduce line voltage. Any transformer used with the controller must have a sufficient power rating for its load. The power rating is expressed in volt-amperes (VA).

Power Supply Inrush

During power-up, the Micro800 power supply allows a brief inrush current to charge internal capacitors. Many power lines and control transformers can supply inrush current for a brief time. If the power source cannot supply this inrush current, the source voltage may sag momentarily.

The only effect of limited inrush current and voltage sag on the Micro800 is that the power supply capacitors charge more slowly. However, the effect of a voltage sag on other equipment should be considered. For example, a deep voltage sag may reset a computer connected to the same power source. The following considerations determine whether the power source must be required to supply high inrush current:

- The power-up sequence of devices in a system.
- The amount of the power source voltage sag if the inrush current cannot be supplied.
- The effect of voltage sag on other equipment in the system.

If the entire system is powered-up at the same time, a brief sag in the power source voltage typically will not affect any equipment.

Loss of Power Source

The optional Micro800 AC power supply is designed to withstand brief power losses without affecting the operation of the system. The time the system is operational during power loss is called program scan hold-up time after loss of power. The duration of the power supply hold-up time depends on power consumption of controller system, but is typically between 10 milliseconds and 3 seconds.

Input States on Power Down

The power supply hold-up time as described above is generally longer than the turn-on and turn-off times of the inputs. Because of this, the input state change from "On" to "Off" that occurs when power is removed may be recorded by the processor before the power supply shuts down the system. Understanding this concept is important. The user program should be written to take this effect into account.

Other Types of Line Conditions

Occasionally the power source to the system can be temporarily interrupted. It is also possible that the voltage level may drop substantially below the normal line voltage range for a period of time. Both of these conditions are considered to be a loss of power for the system.

Preventing Excessive Heat

For most applications, normal convective cooling keeps the controller within the specified operating range. Ensure that the specified temperature range is maintained. Proper spacing of components within an enclosure is usually sufficient for heat dissipation.

In some applications, a substantial amount of heat is produced by other equipment inside or outside the enclosure. In this case, place blower fans inside the enclosure to assist in air circulation and to reduce "hot spots" near the controller.

Additional cooling provisions might be necessary when high ambient temperatures are encountered.

TIP Do not bring in unfiltered outside air. Place the controller in an enclosure to protect it from a corrosive atmosphere. Harmful contaminants or dirt could cause improper operation or damage to components. In extreme cases, you may need to use air conditioning to protect against heat build-up within the enclosure.

Master Control Relay

A hard-wired master control relay (MCR) provides a reliable means for emergency machine shutdown. Since the master control relay allows the placement of several emergency-stop switches in different locations, its installation is important from a safety standpoint. Overtravel limit switches or mushroom-head push buttons are wired in series so that when any of them opens, the master control relay is de-energized. This removes power to input and output device circuits. See illustrations Schematic – Using IEC Symbols on page 29 and Schematic – Using ANSI/CSA Symbols) on page 30.

WARNING: Never alter these circuits to defeat their function since serious injury and/or machine damage could result.

TIP If you are using an external DC power supply, interrupt the DC output side rather than the AC line side of the supply to avoid the additional delay of power supply turn-off.

The AC line of the DC output power supply should be fused.

Connect a set of master control relays in series with the DC power supplying the input and output circuits.

Place the main power disconnect switch where operators and maintenance personnel have quick and easy access to it. If you mount a disconnect switch inside the controller enclosure, place the switch operating handle on the outside of the enclosure, so that you can disconnect power without opening the enclosure.

Whenever any of the emergency-stop switches are opened, power to input and output devices should be removed.

When you use the master control relay to remove power from the external I/O circuits, power continues to be provided to the controller's power supply so that diagnostic indicators on the processor can still be observed.

The master control relay is not a substitute for a disconnect to the controller. It is intended for any situation where the operator must quickly de-energize I/O devices only. When inspecting or installing terminal connections, replacing output fuses, or working on equipment within the enclosure, use the disconnect to shut off power to the rest of the system.

TIP Do not control the master control relay with the controller. Provide the operator with the safety of a direct connection between an emergency-stop switch and the master control relay.

Using Emergency-Stop Switches

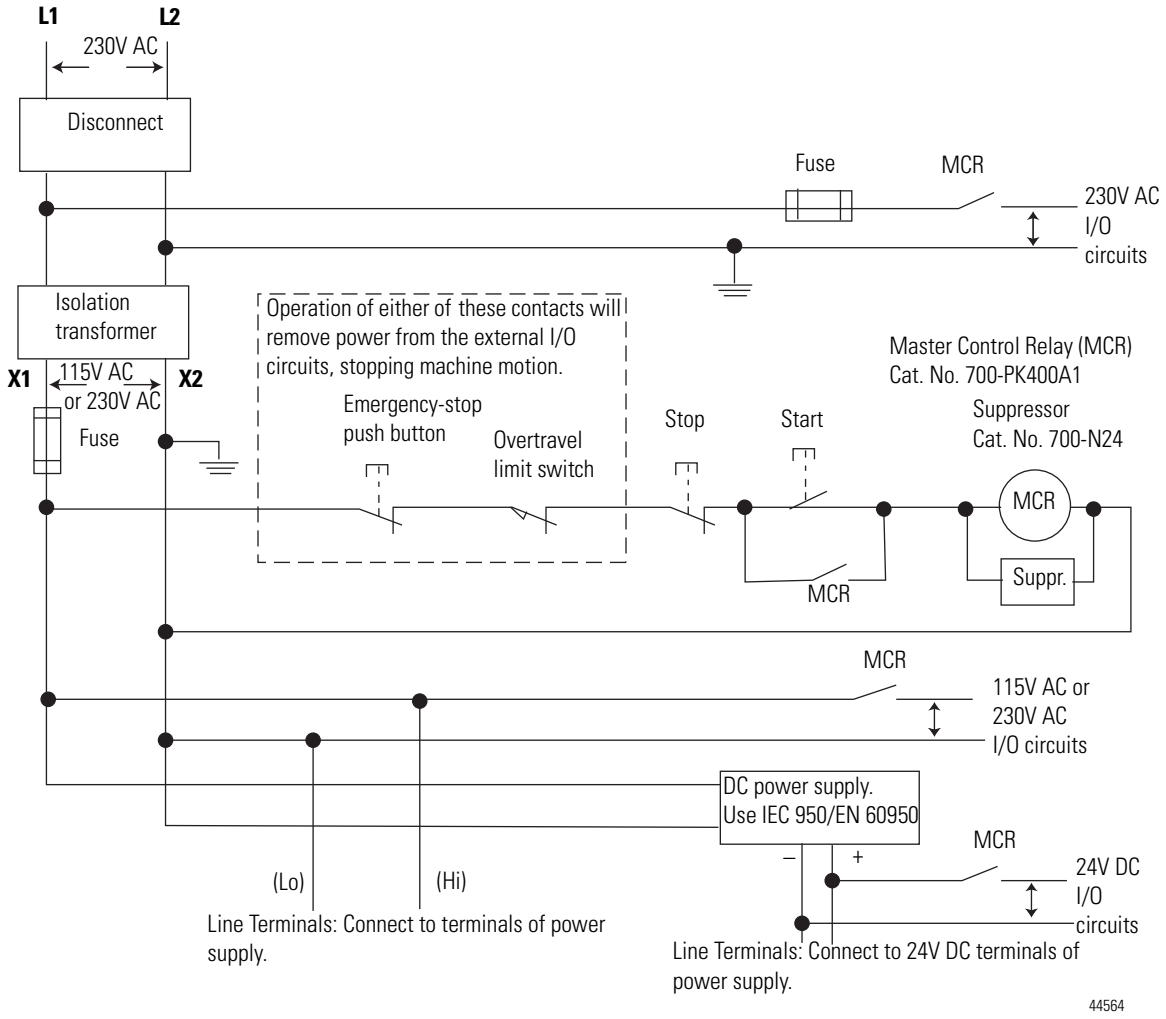
When using emergency-stop switches, adhere to the following points:

- Do not program emergency-stop switches in the controller program. Any emergency-stop switch should turn off all machine power by turning off the master control relay.
- Observe all applicable local codes concerning the placement and labeling of emergency-stop switches.
- Install emergency-stop switches and the master control relay in your system. Make certain that relay contacts have a sufficient rating for your application. Emergency-stop switches must be easy to reach.
- In the following illustration, input and output circuits are shown with MCR protection. However, in most applications, only output circuits require MCR protection.

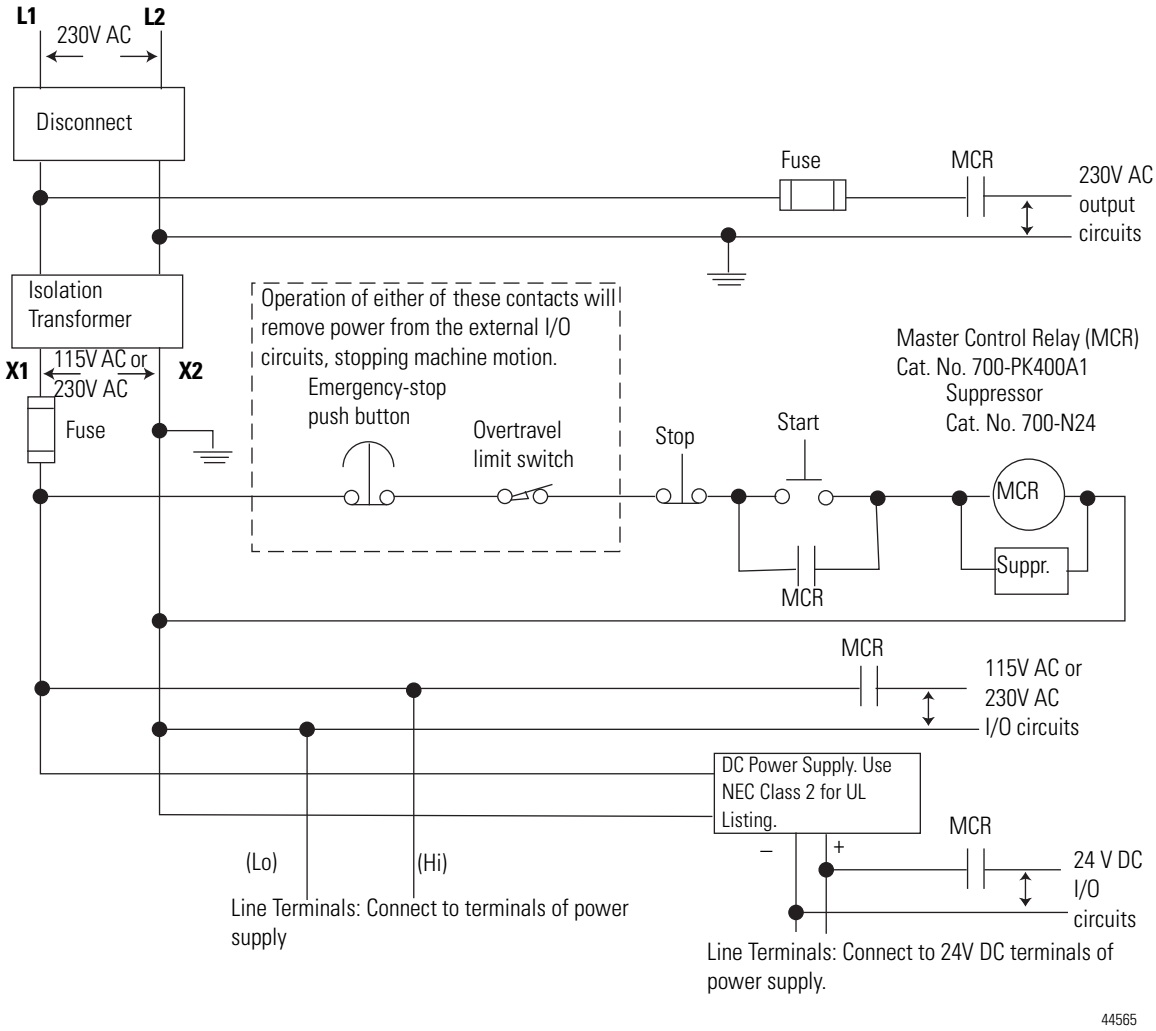
The following illustrations show the Master Control Relay wired in a grounded system.

TIP In most applications input circuits do not require MCR protection; however, if you need to remove power from all field devices, you must include MCR contacts in series with input power wiring.

Schematic – Using IEC Symbols



Schematic – Using ANSI/CSA Symbols)



Install Your Controller

This chapter serves to guide the user on installing the controller. It includes the following topics.

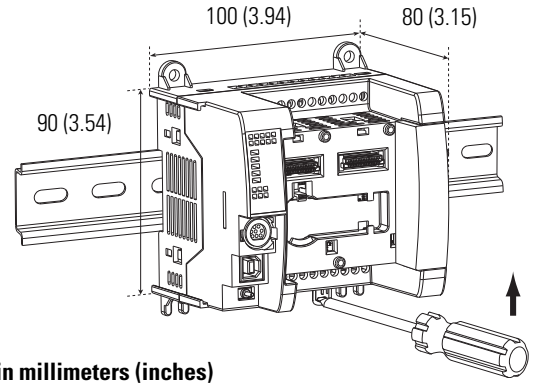
Topic	Page
Controller Mounting Dimensions	31
Mounting Dimensions	31
DIN Rail Mounting	33
Panel Mounting	34

Controller Mounting Dimensions

Mounting Dimensions

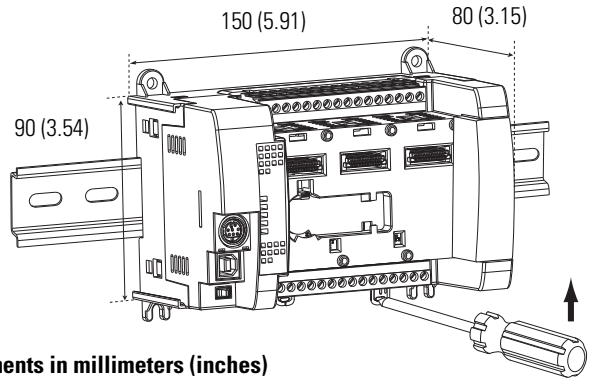
Mounting dimensions do not include mounting feet or DIN rail latches.

Micro830 10- and 16-Point Controllers
2080-LC30-10QWB, 2080-LC30-10QVB,
2080-LC30-16AWB, 2080-LC30-16QWB, 2080-LC30-16QVB



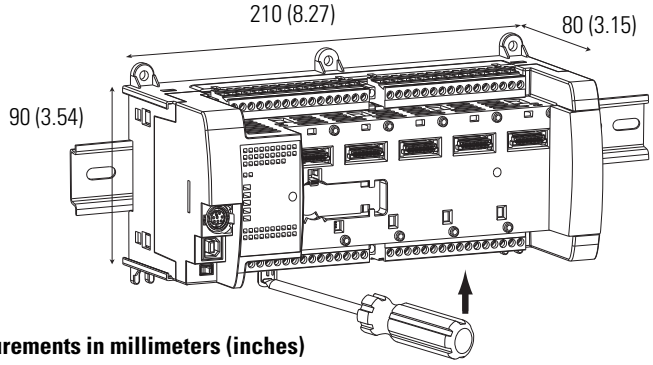
Measurements in millimeters (inches)

Micro830 24-Point Controllers
2080-LC30-24QWB, 2080-LC30-24QVB, 2080-LC30-24QBB



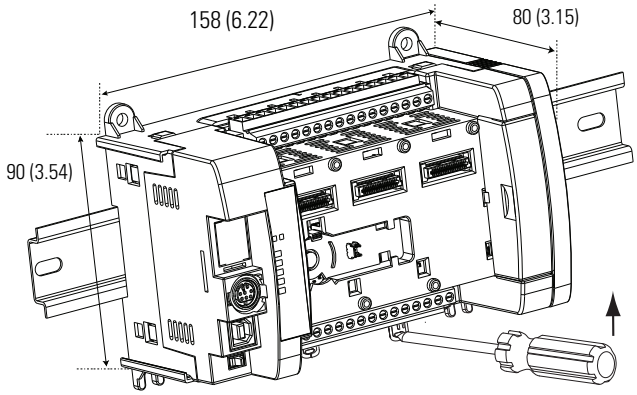
Measurements in millimeters (inches)

Micro830 48-Point Controllers
2080-LC30-48AWB, 2080-LC30-48QWB, 2080-LC30-48QVB, 2080-LC30-48QBB



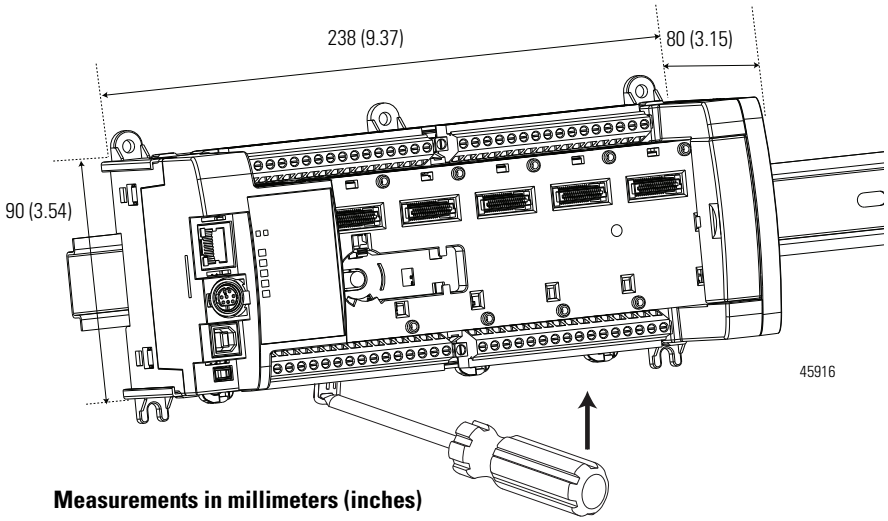
Measurements in millimeters (inches)

Micro850 24-Point Controllers
2080-LC50-24AWB, 2080-LC50-24QBB, 2080-LC50-24QVB, 2080-LC50-24QWB



Measurements in millimeters (inches)

Micro850 48-Point Controllers
2080-LC50-48AWB, 2080-LC50-48QWB, 2080-LC50-48QBB, 2080-LC50-48QVB



Measurements in millimeters (inches)

Maintain spacing from objects such as enclosure walls, wireways and adjacent equipment. Allow 50.8 mm (2 in.) of space on all sides for adequate ventilation. If optional accessories/modules are attached to the controller, such as the power supply 2080-PS120-240VAC or expansion I/O modules, make sure that there is 50.8 mm (2 in.) of space on all sides after attaching the optional parts.

DIN Rail Mounting

The module can be mounted using the following DIN rails: 35 x 7.5 x 1 mm (EN 50 022 - 35 x 7.5).

TIP For environments with greater vibration and shock concerns, use the panel mounting method, instead of DIN rail mounting.

Before mounting the module on a DIN rail, use a flat-blade screwdriver in the DIN rail latch and pry it downwards until it is in the unlatched position.

1. Hook the top of the DIN rail mounting area of the controller onto the DIN rail, and then press the bottom until the controller snaps onto the DIN rail.
2. Push the DIN rail latch back into the latched position.
   Use DIN rail end anchors (Allen-Bradley part number 1492-EAJ35 or 1492-EAHJ35) for vibration or shock environments.

To remove your controller from the DIN rail, pry the DIN rail latch downwards until it is in the unlatched position.

Panel Mounting

The preferred mounting method is to use four M4 (#8) screws per module. Hole spacing tolerance: ±0.4 mm (0.016 in.).

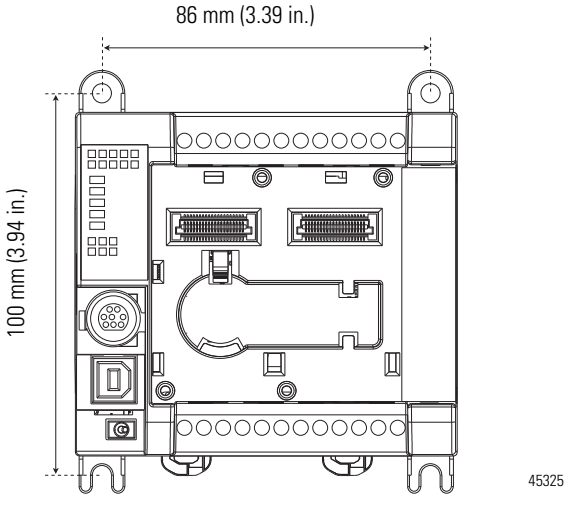
Follow these steps to install your controller using mounting screws.

1. Place the controller against the panel where you are mounting it. Make sure the controller is spaced properly.
2. Mark drilling holes through the mounting screw holes and mounting feet then remove the controller.
3. Drill the holes at the markings, then replace the controller and mount it. Leave the protective debris strip in place until you are finished wiring the controller and any other devices.

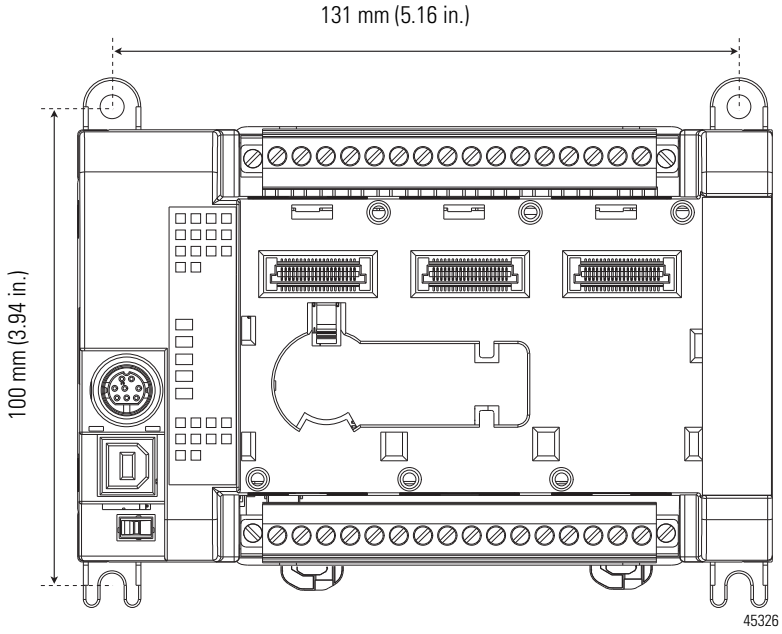
IMPORTANT For instructions on how to install your Micro800 system with expansion I/O, see the User Manual for Micro800 Expansion I/O Modules, 2080-UM003.

Panel Mounting Dimensions

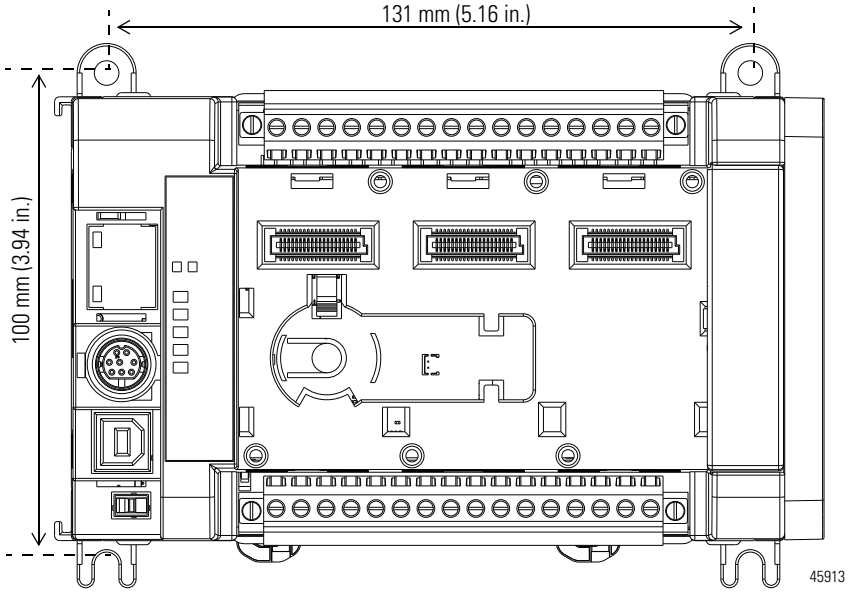
Micro830 10- and 16-Point Controllers
2080-LC30-10QWB, 2080-LC30-10QVB, 2080-LC30-16AWB, 2080-LC30-16QWB, 2080-LC30-16QVB



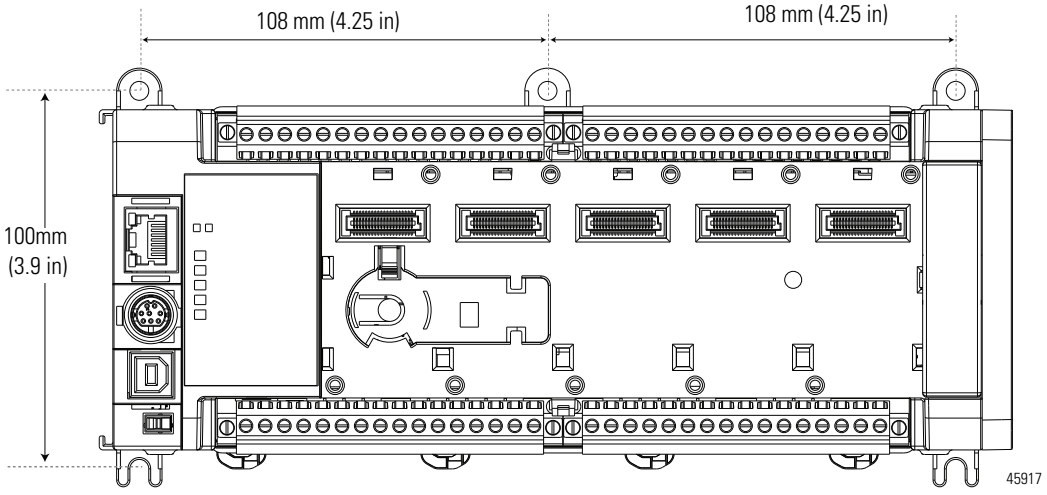
Micro830 24-Point Controllers
2080-LC30-24QWB, 2080-LC30-24QVB, 2080-LC30-24QBB



Micro850 24-Point Controllers
2080-LC50-24AWB, 2080-LC50-24QBB, 2080-LC50-24QVB, 2080-LC50-24QWB

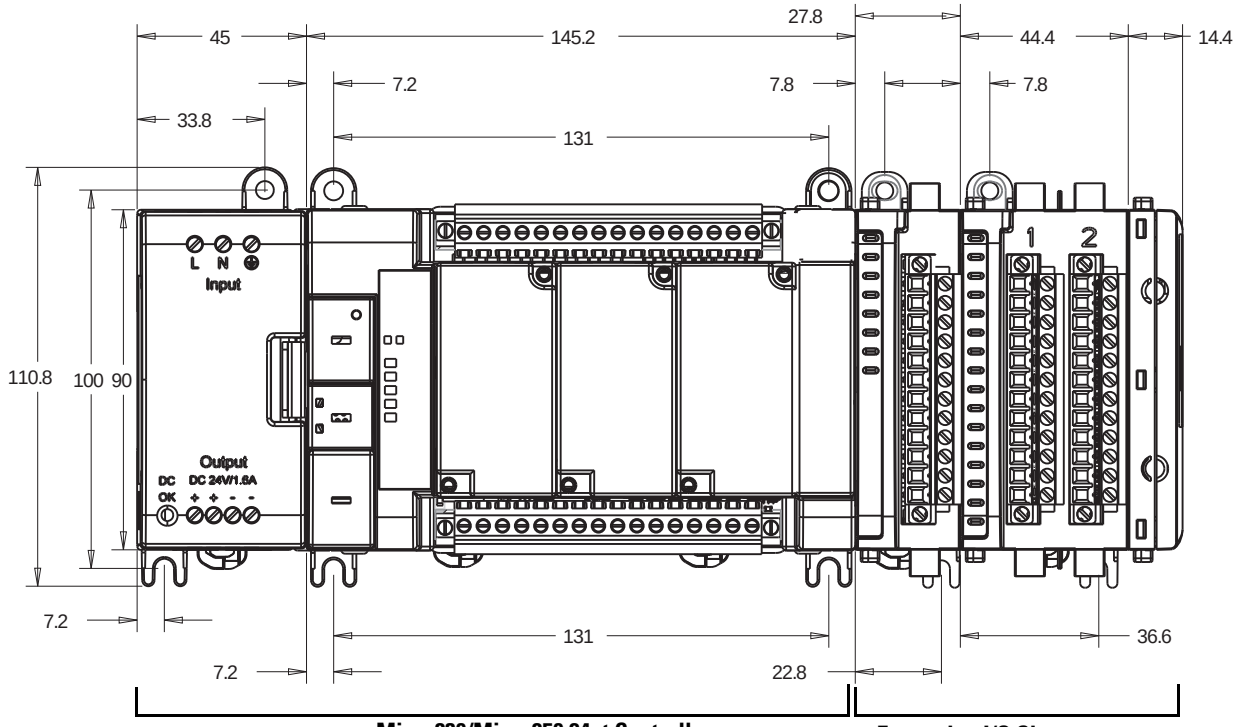


Micro830 48-Point Controllers
2080-LC30-48AWB, 2080-LC30-48QWB, 2080-LC30-48QVB, 2080-LC30-48QBB


108 mm (4.25 in)
108 mm (4.25 in)
100mm
(3.9 in)
45917

System Assembly

Micro830 and Micro850 24-point Controllers (Front)

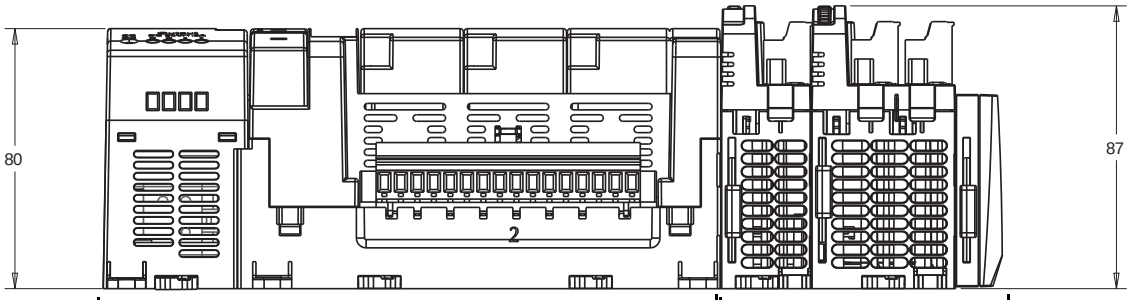


Micro830/Micro850 24pt Controller with Micro800 Power Supply

Expansion I/O Slots
(Applicable to Micro850 only)
Single-width (1st slot)
Double-width (2nd slot)
2085-ECR (terminator)

Measurements in millimeters

Micro830 and Micro850 24-point Controllers (Side)

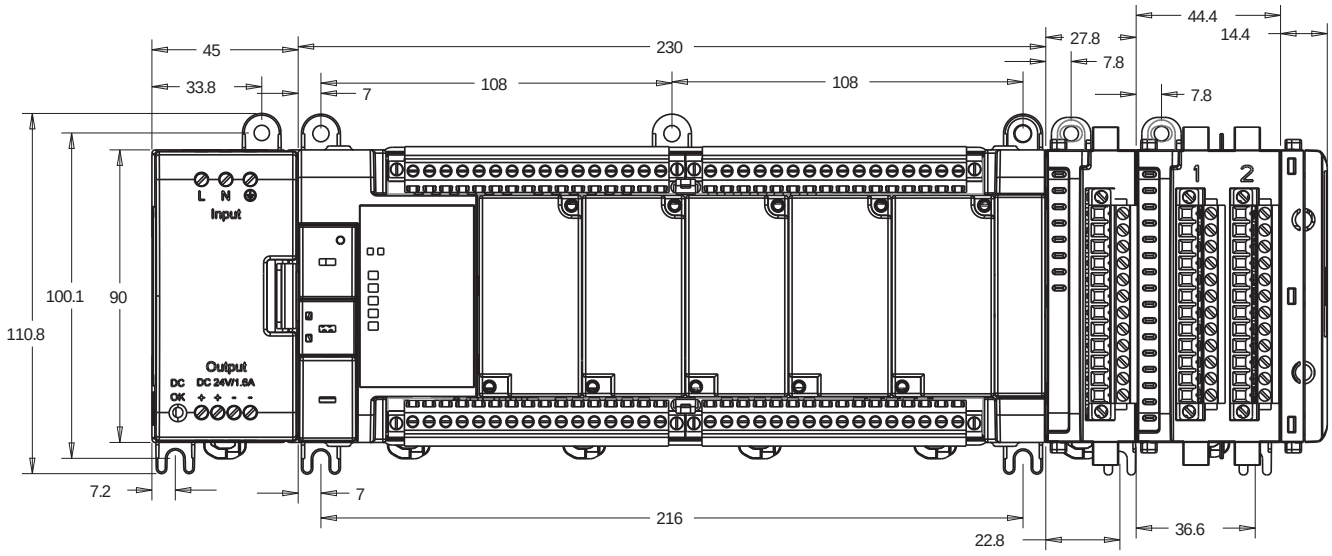


Micro830/Micro850 24pt Controller with Micro800 Power Supply

Expansion I/O Slots
(Applicable to Micro850 only)
Single-width (1st slot)
Double-width (2nd slot)
2085-ECR (terminator)

Measurements in millimeters

Micro830 and Micro850 48-point Controllers (Front)

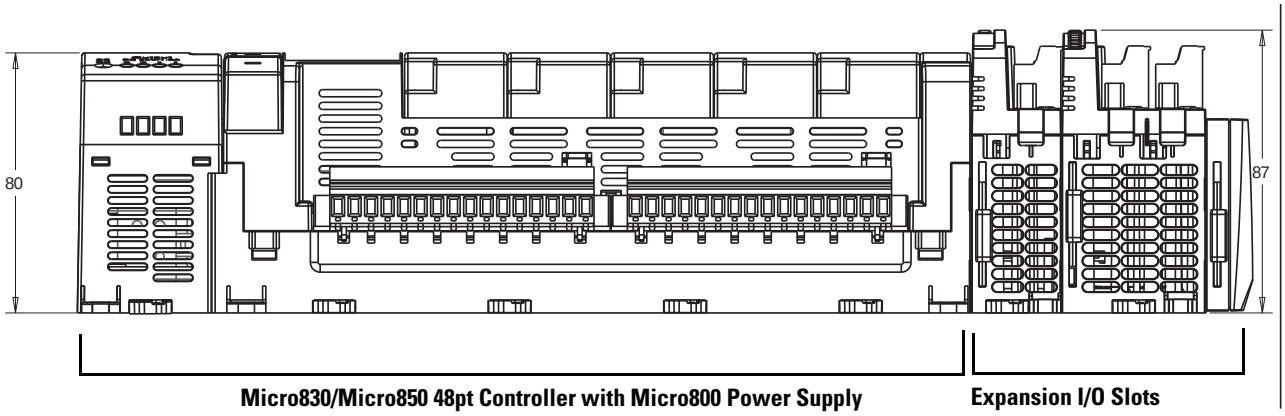


Micro830/Micro850 48pt Controller with Micro800 Power Supply

Expansion I/O Slots
(Applicable to Micro850 only)
Single-width (1st slot)
Double-width (2nd slot)
2085-ECR (terminator)

Measurements in millimeters

Micro830 and Micro850 48-point Controllers (Side)



Micro830/Micro850 48pt Controller with Micro800 Power Supply

Expansion I/O Slots
(Applicable to Micro850 only)
Single-width (1st slot)
Double-width (2nd slot)
2085-ECR (terminator)

Measurements in millimeters

Wire Your Controller

This chapter provides information on the Micro830 and Micro850 controller wiring requirements. It includes the following sections:

Topic	Page
Wiring Requirements and Recommendation	39
Use Surge Suppressors	40
Recommended Surge Suppressors	42
Grounding the Controller	43
Wiring Diagrams	43
Controller I/O Wiring	46
Minimize Electrical Noise	47
Analog Channel Wiring Guidelines	47
Minimize Electrical Noise on Analog Channels	47
Grounding Your Analog Cable	48
Wiring Examples	48
Embedded Serial Port Wiring	49

Wiring Requirements and Recommendation

WARNING: Before you install and wire any device, disconnect power to the controller system.

WARNING: Calculate the maximum possible current in each power and common wire. Observe all electrical codes dictating the maximum current allowable for each wire size. Current above the maximum ratings may cause wiring to overheat, which can cause damage.
United States Only: If the controller is installed within a potentially hazardous environment, all wiring must comply with the requirements stated in the National Electrical Code 501-10 (b).

- Allow for at least 50 mm (2 in.) between I/O wiring ducts or terminal strips and the controller.

- Route incoming power to the controller by a path separate from the device wiring. Where paths must cross, their intersection should be perpendicular.

  TIP Do not run signal or communications wiring and power wiring in the same conduit. Wires with different signal characteristics should be routed by separate paths.

- Separate wiring by signal type. Bundle wiring with similar electrical characteristics together.
- Separate input wiring from output wiring.
- Label wiring to all devices in the system. Use tape, shrink-tubing, or other dependable means for labeling purposes. In addition to labeling, use colored insulation to identify wiring based on signal characteristics. For example, you may use blue for DC wiring and red for AC wiring.

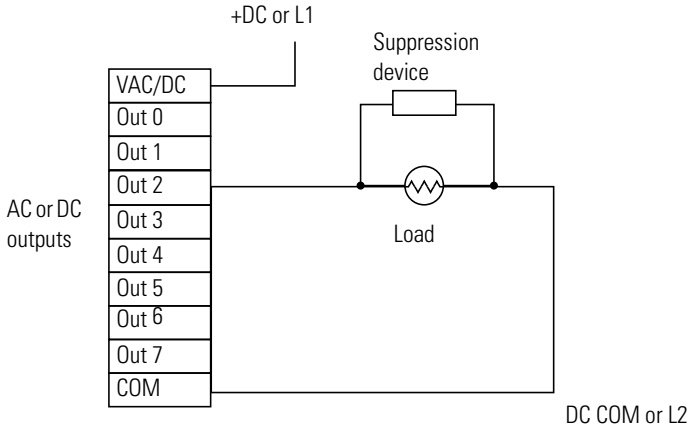
Wire Requirements

	Wire Size			
	Type	Min	Max	
Micro830/ Micro850 Controllers	Solid	0.2 mm^2 (24 AWG)	2.5 mm^2 (12 AWG)	rated @ 90 °C (194 °F) insulation max
	Stranded	0.2 mm^2 (24 AWG)	2.5 mm^2 (12 AWG)	

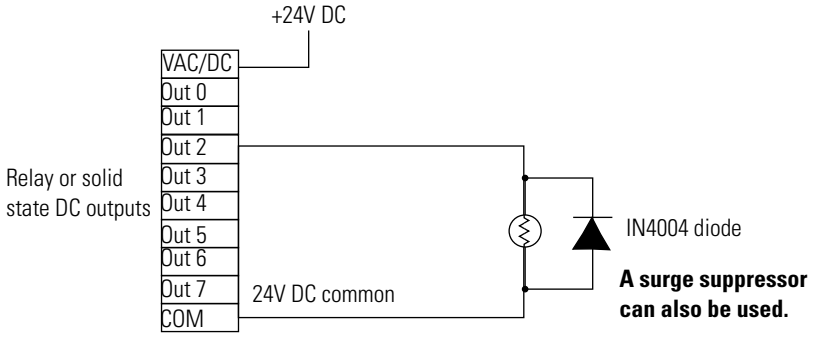
Use Surge Suppressors

Because of the potentially high current surges that occur when switching inductive load devices, such as motor starters and solenoids, the use of some type of surge suppression to protect and extend the operating life of the controllers output contacts is required. Switching inductive loads without surge suppression can *significantly* reduce the life expectancy of relay contacts. By adding a suppression device directly across the coil of an inductive device, you prolong the life of the output or relay contacts. You also reduce the effects of voltage transients and electrical noise from radiating into adjacent systems.

The following diagram shows an output with a suppression device. We recommend that you locate the suppression device as close as possible to the load device.



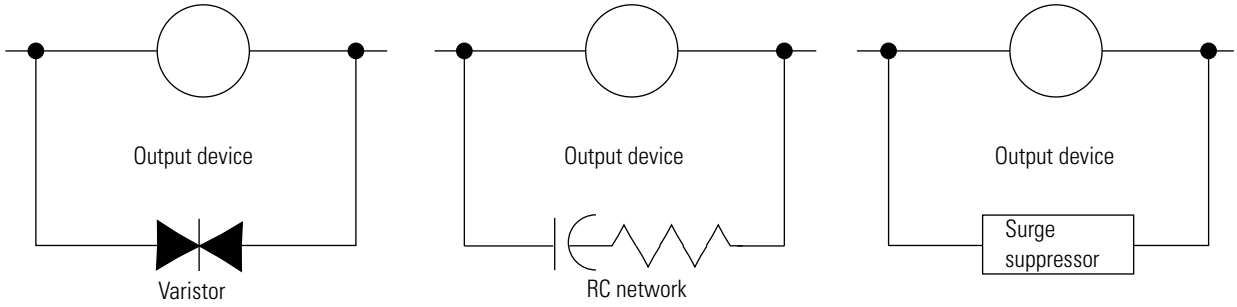
If the outputs are DC, we recommend that you use an 1N4004 diode for surge suppression, as shown below. For inductive DC load devices, a diode is suitable. A 1N4004 diode is acceptable for most applications. A surge suppressor can also be used. See Recommended Surge Suppressors on page 42. As shown below, these surge suppression circuits connect directly across the load device.



Suitable surge suppression methods for inductive AC load devices include a varistor, an RC network, or an Allen-Bradley surge suppressor, all shown below. These components must be appropriately rated to suppress the switching

transient characteristic of the particular inductive device. See Recommended Surge Suppressors on page 42 for recommended suppressors.

Surge Suppression for Inductive AC Load Devices



Recommended Surge Suppressors

Use the Allen-Bradley surge suppressors shown in the following table for use with relays, contactors, and starters.

Recommended Surge Suppressors

Device	Coil Voltage	Suppressor Catalog Number	Type[4]
Bulletin 100/104K 700K	24…48V AC	100-KFSC50	RC
	110…280V AC	100-KFSC280	
	380…480V AC	100-KFSC480	
	12…55 V AC, 12…77V DC	100-KFSV55	MOV
	56…136 VAC, 78…180V DC	100-KFSV136	
	137…277V AC, 181…250 V DC	100-KFSV277	
	12…250V DC	100-KFSD250	Diode
Bulletin 100C, (C09 - C97)	24…48V AC	100-FSC48[1]	RC
	110…280V AC	100-FSC280[1]	
	380…480V AC	100-FSC480[1]	
	12…55V AC, 12…77V DC	100-FSV55[1]	MOV
	56…136V AC, 78…180V DC	100-FSV136[1]	
	137…277V AC, 181…250V DC	100-FSV277[1]	
	278…575V AC	100-FSV575[1]	
	12…250V DC	100-FSD250[1]	Diode
Bulletin 509 Motor Starter Size 0 - 5	12…120V AC	599-K04	MOV
	240…264V AC	599-KA04	

Recommended Surge Suppressors

Device	Coil Voltage	Suppressor Catalog Number	Type[4]
Bulletin 509 Motor Starter Size 6	12…120V AC	199-FSMA1[2]	RC
	12…120V AC	199-GSMA1[3]	MOV
Bulletin 700 R/RM Relay	AC coil	Not Required	
	24…48V DC	199-FSMA9	MOV
	50…120V DC	199-FSMA10	
	130…250V DC	199-FSMA11	
Bulletin 700 Type N, P, PK or PH Relay	6…150V AC/DC	700-N24	RC
	24…48V AC/DC	199-FSMA9	MOV
	50…120V AC/DC	199-FSMA10	
	130…250V AC/DC	199-FSMA11	
	6…300V DC	199-FSMZ-1	Diode
Miscellaneous electromagnetic devices limited to 35 sealed VA	6…150V AC/DC	700-N24	RC

(1) Catalog numbers for screwless terminals include the string 'CR' after '100-'. For example: Cat. No. 100-FSC48 becomes Cat. No. 100-**CR**FSC48; Cat. No. 100-FSV55 becomes 100-**CR**FSV55; and so on.

(2) For use on the interposing relay.

(3) For use on the contactor or starter.

(4) RC Type not to be used with Triac outputs. Varistor is not recommended for use on the relay outputs.

Grounding the Controller

WARNING: All devices connected to the RS-232/485 communication port must be referenced to controller ground, or be floating (not referenced to a potential other than ground). Failure to follow this procedure may result in property damage or personal injury.

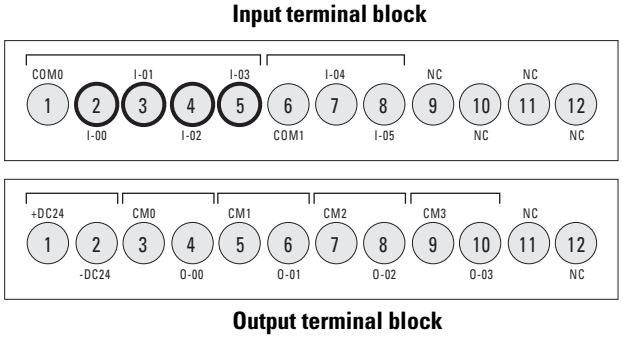
This product is intended to be mounted to a well grounded mounting surface such as a metal panel. Refer to the Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1, for additional information.

Wiring Diagrams

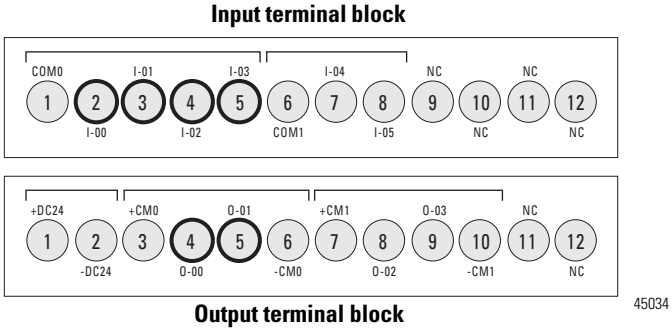
The following illustrations show the wiring diagrams for the Micro800 controllers. Controllers with DC inputs can be wired as either sinking or sourcing inputs. Sinking and sourcing does not apply to AC inputs.

High-speed inputs and outputs are indicated by ○.

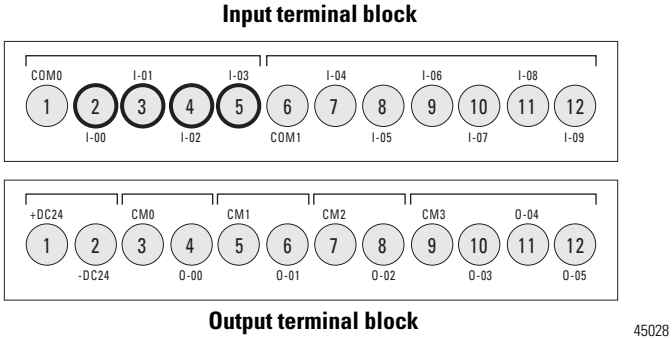
2080-LC30-10QWB


Input terminal block
COM0 I-01 I-03 I-04 NC NC
1 2 3 4 5 6 7 8 9 10 11 12
I-00 I-02 COM1 I-05 NC NC
+DC24 CM0 CM1 CM2 CM3 NC
1 2 3 4 5 6 7 8 9 10 11 12
-DC24 O-00 O-01 O-02 O-03 NC
Output terminal block
45033

2080-LC30-10QVB

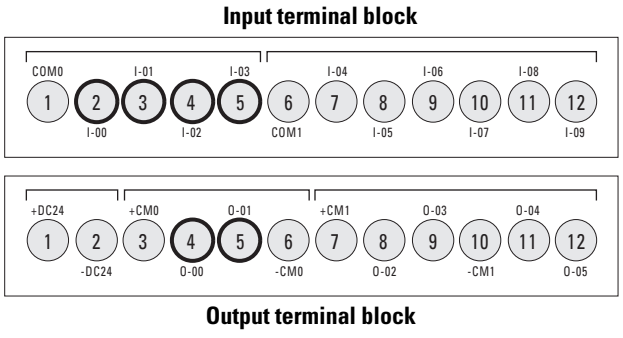

Input terminal block
COM0 I-01 I-03 I-04 NC NC
1 2 3 4 5 6 7 8 9 10 11 12
I-00 I-02 COM1 I-05 NC NC
+DC24 +CM0 O-01 +CM1 O-03 NC
1 2 3 4 5 6 7 8 9 10 11 12
-DC24 O-00 -CM0 O-02 -CM1 NC
Output terminal block
45034

2080-LC30-16AWB / 2080-LC30-16QWB

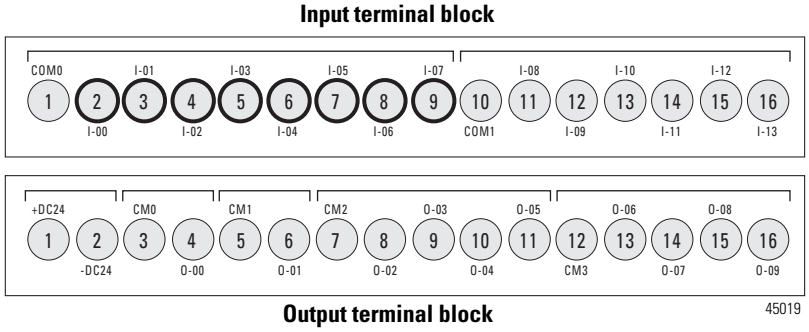


TIP 2080-LC30-16AWB has no high-speed inputs.

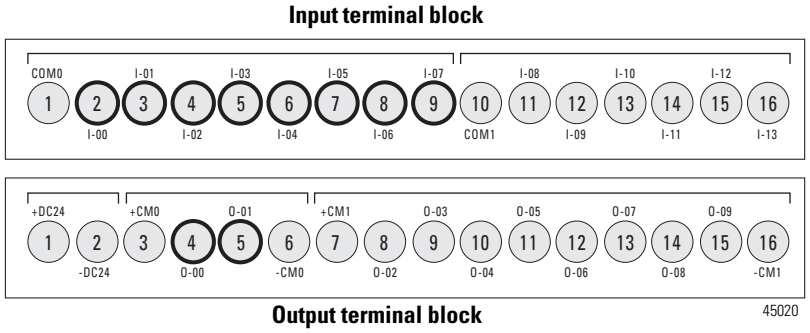
2080-LC30-16QVB



2080-LC30-24QWB / 2080-LC50-24AWB / 2080-LC50-24QWB

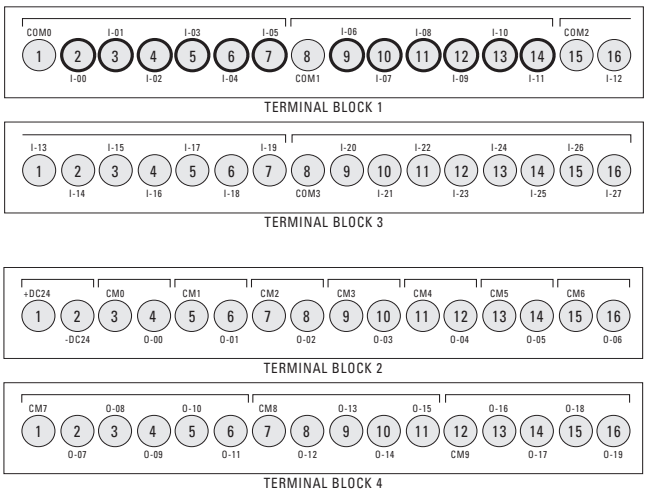


2080-LC30-24QVB / 2080-LC30-24QBB / 2080-LC50-24QVB / 2080-LC50-24QBB



2080-LC30-48AWB / 2080-LC30-48QWB / 2080-LC50-48AWB / 2080-LC50-48QWB

Input terminal block



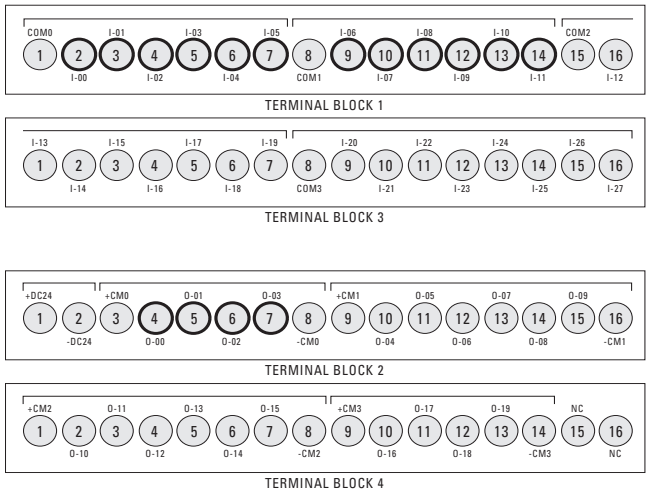
45039

Output terminal block

TIP 2080-LC30-48AWB has no high-speed inputs.

2080-LC30-48QVB / 2080-LC30-48QBB / 2080-LC50-48QVB / 2080-LC50-48QBB

Input terminal block



45040

Output terminal block

Controller I/O Wiring

This section contains some relevant information about minimizing electrical noise and also includes some wiring examples.

Minimize Electrical Noise

Because of the variety of applications and environments where controllers are installed and operating, it is impossible to ensure that all environmental noise will be removed by input filters. To help reduce the effects of environmental noise, install the Micro800 system in a properly rated (for example, NEMA) enclosure. Make sure that the Micro800 system is properly grounded.

A system may malfunction due to a change in the operating environment after a period of time. We recommend periodically checking system operation, particularly when new machinery or other noise sources are installed near the Micro800 system.

Analog Channel Wiring Guidelines

Consider the following when wiring your analog channels:

- The analog common (COM) is not electrically isolated from the system, and is connected to the power supply common.
- Analog channels are not isolated from each other.
- Use Belden cable #8761, or equivalent, shielded wire.
- Under normal conditions, the drain wire (shield) should be connected to the metal mounting panel (earth ground). Keep the shield connection to earth ground as short as possible.
- To ensure optimum accuracy for voltage type inputs, limit overall cable impedance by keeping all analog cables as short as possible. Locate the I/O system as close to your voltage type sensors or actuators as possible.

Minimize Electrical Noise on Analog Channels

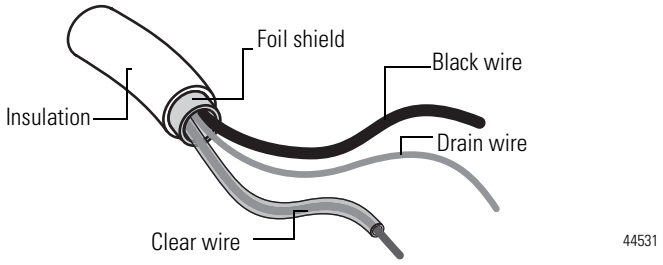
Inputs on analog channels employ digital high-frequency filters that significantly reduce the effects of electrical noise on input signals. However, because of the variety of applications and environments where analog controllers are installed and operated, it is impossible to ensure that all environmental noise will be removed by the input filters.

Several specific steps can be taken to help reduce the effects of environmental noise on analog signals:

- install the Micro800 system in a properly rated enclosure, for example, NEMA. Make sure that the shield is properly grounded.
- use Belden cable #8761 for wiring the analog channels, making sure that the drain wire and foil shield are properly earth grounded.
- route the Belden cable separately from any AC wiring. Additional noise immunity can be obtained by routing the cables in grounded conduit.

Grounding Your Analog Cable

Use shielded communication cable (Belden #8761). The Belden cable has two signal wires (black and clear), one drain wire, and a foil shield. The drain wire and foil shield must be grounded at one end of the cable.

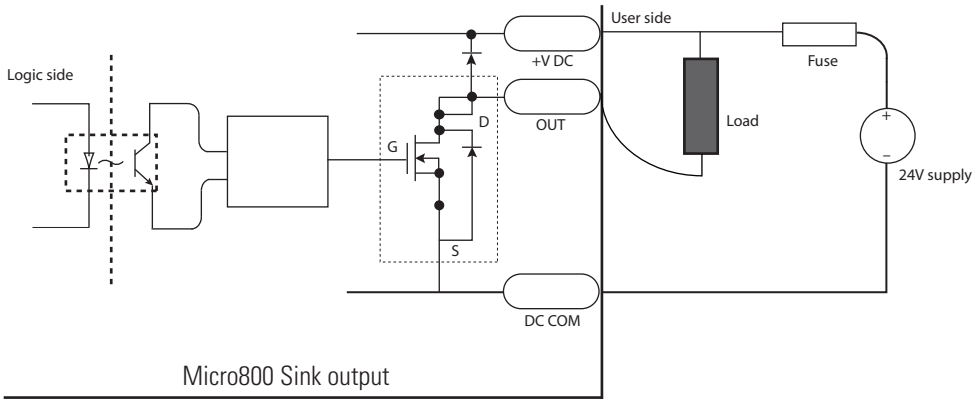


IMPORTANT Do not ground the drain wire and foil shield at both ends of the cable.

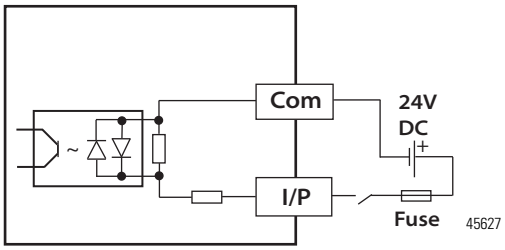
Wiring Examples

Examples of sink/source, input/output wiring are shown below.

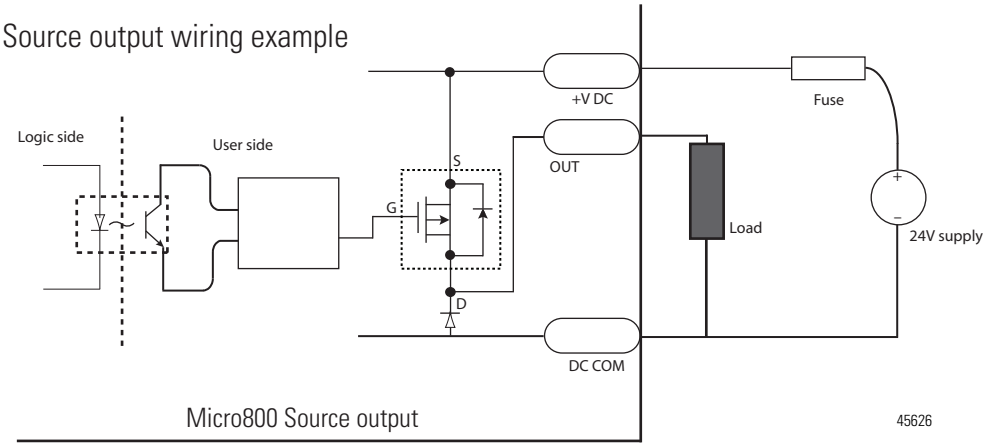
Sink output wiring example



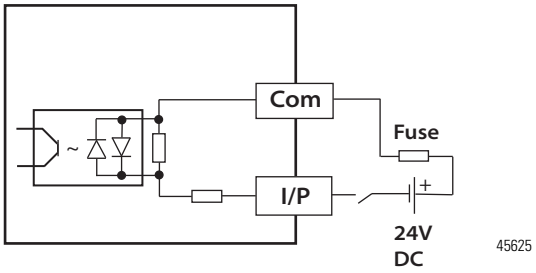
Sink input wiring example



Source output wiring example



Source input wiring example



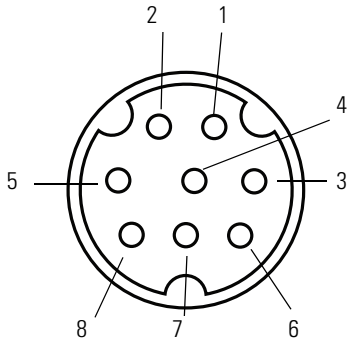
Embedded Serial Port Wiring

The embedded serial port is a non-isolated RS232/RS485 serial port which is targeted to be used for short distances (<3 m) to devices such as HMIs.

See Embedded Serial Port Cables on page 7 for a list of cables that can be used with the embedded serial port 8-pin Mini DIN connector.

For example the 1761-CBL-PM02 cable is typically used to connect the embedded serial port to PanelView Component HMI using RS232.

Embedded Serial Port



Pinout table

Pin	Definition	RS-485 Example	RS-232 Example
1	RS-485+	B(+)	(not used)
2	GND	GND	GND
3	RS-232 RTS	(not used)	RTS
4	RS-232 RxD	(not used)	RxD
5	RS-232 DCD	(not used)	DCD
6	RS-232 CTS	(not used)	CTS
7	RS-232 TxD	(not used)	TxD
8	RS-485-	A(-)	(not used)

Communication Connections

Overview

This chapter describes how to communicate with your control system and configure communication settings. The method you use and cabling required to connect your controller depends on what type of system you are employing. This chapter also describes how the controller establishes communication with the appropriate network. Topics include:

Topic	Page
Supported Communication Protocols	51
Use Modems with Micro800 Controllers	57
Configure Serial Port	58
Configure Ethernet Settings	64
OPC Support Using RSLinx Enterprise	67

The Micro830 and Micro850 controllers have the following embedded communication channels:

- a non-isolated RS-232/RS-485 combo port
- a non-isolated USB programming port

In addition, the Micro850 controller has an RJ-45 Ethernet port.

Supported Communication Protocols

Micro830/Micro850 controllers support communication through the embedded RS-232/RS-485 serial port as well as any installed serial port plug-in modules. In addition, Micro850 controllers also support communication through the embedded Ethernet port, and can be connected to a local area network for various devices providing 10 Mbps/100 Mbps transfer rate.

These are the communication protocols supported by Micro830/Micro850 controllers:

- Modbus RTU Master and Slave
- CIP Serial Client/Server (RS-232 only)
- CIP Symbolic Client/Server
- ASCII

These are the communication protocols supported by Micro850 controllers only:

- EtherNet/IP Client/Server
- Modbus TCP Client/Server
- DHCP Client

- Sockets Client/Server TCP/UDP

Connection Limits for Micro830/Micro850 Controllers

Description		Micro830	Micro850
CIP Connections			
Total number of client plus server connections for all ports		16	24
Maximum number of client connections for all ports		15	16
Maximum number of server connections for all ports		16	24
Maximum number of EtherNet/IP connections	Client	–	16
	Server		23
Maximum number of USB connections	Client	–	–
	Server	15	23
Maximum number of Serial connections	Client	15	16
	Server	15	23
TCP Connections			
Total number of client plus server connections		–	64
Maximum number for EtherNet/IP	Client		16
	Server		16
Maximum number for Modbus TCP	Client		16
	Server		16
Maximum number for User Programmable Sockets			8
User Programmable Sockets			
Total number of User Programmable Sockets (any combination of UDP plus TCP Client/Server)		–	8

IMPORTANT If all client/server connections are fully loaded, performance may be affected, such as data loss and intermittent delays during communication.

Here are some configuration examples based on the limits described in the table above:

1. The maximum number of drives that can be controlled over EtherNet/IP is 16. This is due to the maximum limit of TCP Client connections is 16, and the maximum limit of EtherNet/IP Client connections is also 16.
2. If you have 10 devices controlled over EtherNet/IP, the maximum number of devices that can be controlled over Serial is six. This is due to the maximum limit of Client connections is 16.
3. The total number of UDP sockets plus TCP Client/Server sockets has a maximum limit of eight.

Modbus RTU

Modbus is a half-duplex, master-slave communications protocol. The Modbus network master reads and writes bits and registers. Modbus protocol allows a single master to communicate with a maximum of 247 slave devices. Micro800 controllers support Modbus RTU Master and Modbus RTU Slave protocol. For more information on configuring your Micro800 controller for Modbus protocol, refer to the Connected Components Workbench Online Help. For more information about the Modbus protocol, refer to the Modbus Protocol Specifications (available from http://www.modbus.org).

See Modbus Mapping for Micro800 on page 195 for information on Modbus mapping. To configure the Serial port as Modbus RTU, see Configure Modbus RTU on page 61.

TIP Use MSG_MODBUS instruction to send Modbus messages over serial port.

CIP Serial Client/Server (RS-232 only)

CIP Serial Client/Server allows CIP protocol to be used over an RS-232 serial port. It is typically used with modems. The advantage over non-CIP serial protocols is that since the protocol is CIP, program downloads are supported including CIP pass-through from the serial port to Ethernet.

ASCII

ASCII provides connection to other ASCII devices, such as bar code readers, weigh scales, serial printers, and other intelligent devices. You can use ASCII by configuring the embedded or any plug-in serial RS232/RS485 port for the ASCII driver. Refer to the Connected Components Workbench Online Help for more information.

To configure the serial port for ASCII, see Configure ASCII on page 62.

Modbus TCP Client/Server

The Modbus TCP Client/Server communication protocol uses the same Modbus mapping features as Modbus RTU, but instead of the Serial port, it is supported over Ethernet. Modbus TCP Server takes on Modbus Slave features on Ethernet.

No protocol configuration is required other than configuring the Modbus mapping table. For information on Modbus mapping, see Modbus Mapping for Micro800 on page 195.

TIP Use MSG_MODBUS2 instruction to send Modbus TCP message over Ethernet port.

CIP Symbolic Client/Server

CIP Symbolic is supported by any CIP compliant interface including Ethernet (EtherNet/IP) and Serial Port (CIP Serial). This protocol allows HMIs to easily connect to the Micro830/Micro850 controller.

Micro850 controllers support up to 16 simultaneous EtherNet/IP Client connections and 23 simultaneous EtherNet/IP Server connections.

CIP Serial, supported on both Micro830 and Micro850 controllers, makes use of DF1 Full Duplex protocol, which provides point-to-point connection between two devices.

The Micro800 controllers support the protocol through RS-232 connection to external devices, such as computers running RSLinx Classic software, PanelView Component terminals (firmware revisions 1.70 and above), PanelView 800 terminals or other controllers that support CIP Serial over DF1 Full-Duplex, such as ControlLogix and CompactLogix controllers that have embedded serial ports.

EtherNet/IP, supported on the Micro850 controller, makes use of the standard Ethernet TCP/IP protocol.

The Micro850 controller supports up to 23 simultaneous EtherNet/IP Server connections.

To configure CIP Serial, see Configure CIP Serial Driver on page 59.

To configure for EtherNet/IP, see Configure Ethernet Settings on page 64.

CIP Symbolic Addressing

Users may access any global variables through CIP Symbolic addressing except for system and reserved variables.

One- or two-dimension arrays for simple data types are supported (for example, ARRAY OF INT[1..10, 1..10]) are supported but arrays of arrays (for example, ARRAY OF ARRAY) are not supported. Array of strings are also supported.

Supported Data Types in CIP Symbolic

Data Type[1]	Description
BOOL	Logical Boolean with values TRUE(1) and FALSE(0) (Uses up 8 bits of memory)
SINT	Signed 8-bit integer value
INT	Signed 16-bit integer value
DINT	Signed 32-bit integer value
LINT[2]	Signed 64-bit integer value
USINT	Unsigned 8-bit integer value
UINT	Unsigned 16-bit integer value
UDINT	Unsigned 32-bit integer value
ULINT[2]	Unsigned 64-bit integer value
REAL	32-bit floating point value
LREAL[2]	64-bit floating point value
STRING	character string (1 byte per character)
DATE[3]	Unsigned 32-bit integer value
TIME[3]	Unsigned 32-bit integer value

(1) Logix MSG instruction can read/write SINT, INT, DINT, LINT and REAL data types using "CIP Data Table Read" and "CIP Data Table Write" message types.
BOOL, USINT, UINT, UDINT, ULINT, LREAL, STRING, SHORT_STRING, DATE, and TIME data types are not accessible with the Logix MSG instruction.

(2) Not supported in PanelView Component or PanelView 800.

(3) Can be used by sending data to UDINT, mainly for use with PanelView Plus and PanelView 800 HMI terminals.

CIP Client Messaging

CIP Generic and CIP Symbolic messages are supported on Micro800 controllers through the Ethernet and serial ports. These client messaging features are enabled by the MSG_CIPSYMBOLIC and MSG_CIPGENERIC function blocks.

See Micro800 Programmable Controllers: Getting Started with CIP Client Messaging, publication 2080-QS002, for more information and sample quickstart projects to help you use the CIP Client Messaging feature.

Sockets Client/Server TCP/UDP

Sockets protocol is used for Ethernet communications to devices which do not support Modbus TCP and Ethernet/IP. Sockets support client and server, and

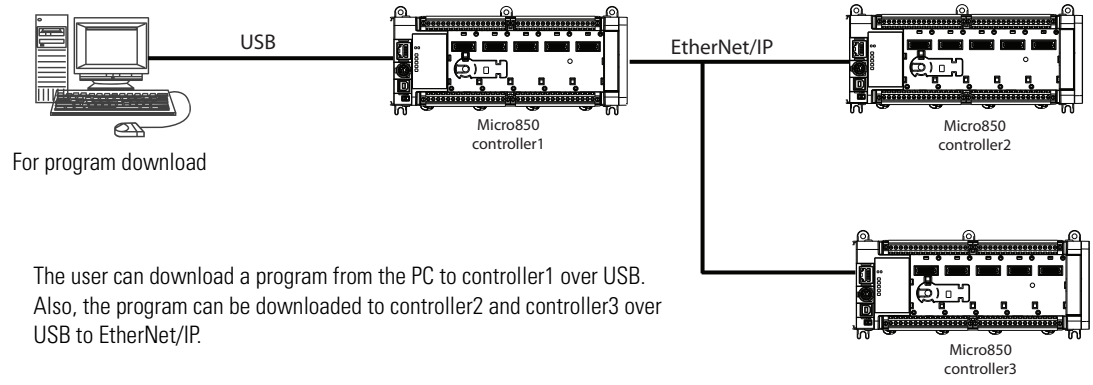
TCP and UDP. Typical applications include communicating to printers, barcode readers, and PCs.

CIP Communications Pass-thru

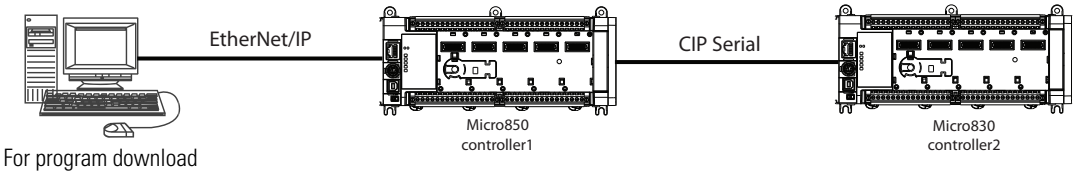
The Micro830 and Micro850 controllers support pass-thru on any communications port that supports Common Industrial Protocol (CIP) for applications such as program download. It does not support applications that require dedicated connections such as HMI. Micro830 and Micro850 support a maximum of one hop. A hop is defined to be an intermediate connection or communications link between two devices – in Micro800, this is through EtherNet/IP or CIP Serial or CIP USB.

Examples of Supported Architectures

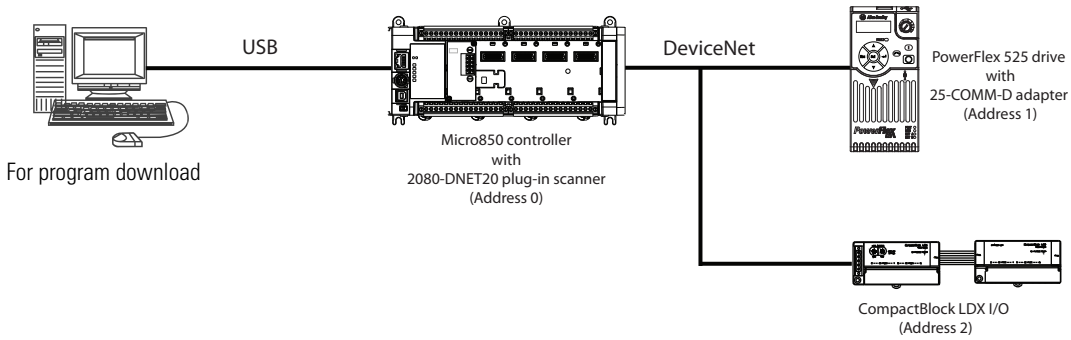
USB to EtherNet/IP



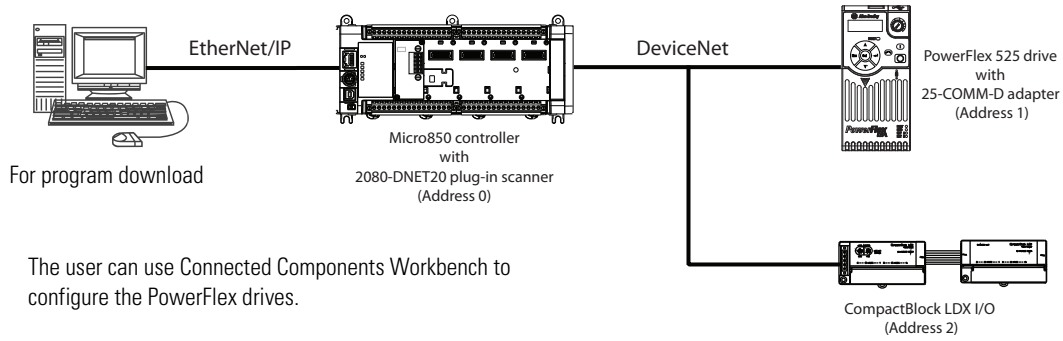
EtherNet/IP to CIP Serial



USB to DeviceNet



EtherNet/IP to DeviceNet



The user can use Connected Components Workbench to configure the PowerFlex drives.

IMPORTANT Micro800 controllers do not support more than one hop (for example, from EtherNet/IP → CIP Serial → EtherNet/IP).

Use Modems with Micro800 Controllers

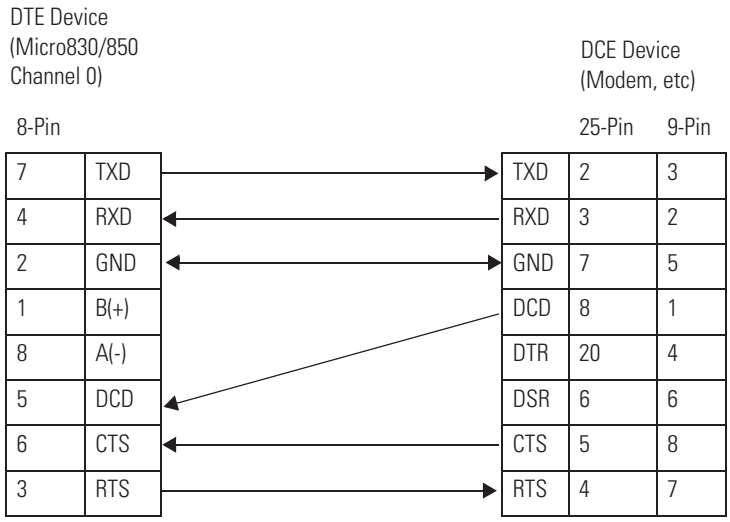
Serial modems can be used with the Micro830 and Micro850 controllers.

Making a DF1 Point-to-Point Connection

You can connect the Micro830 and Micro850 programmable controller to your serial modem using an Allen-Bradley null modem serial cable (1761-CBL-PM02) to the controller's embedded serial port together with a 9-pin null modem adapter – a null modem with a null modem adapter is equivalent to a modem cable. The recommended protocol for this configuration is CIP Serial.

Construct Your Own Modem Cable

If you construct your own modem cable, the maximum cable length is 15.24 m (50 ft) with a 25-pin or 9-pin connector. Refer to the following typical pinout for constructing a straight-through cable:

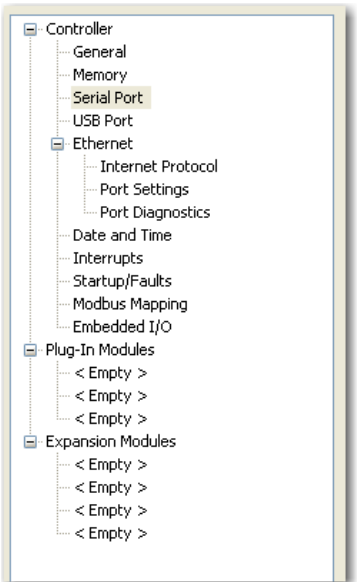


Configure Serial Port

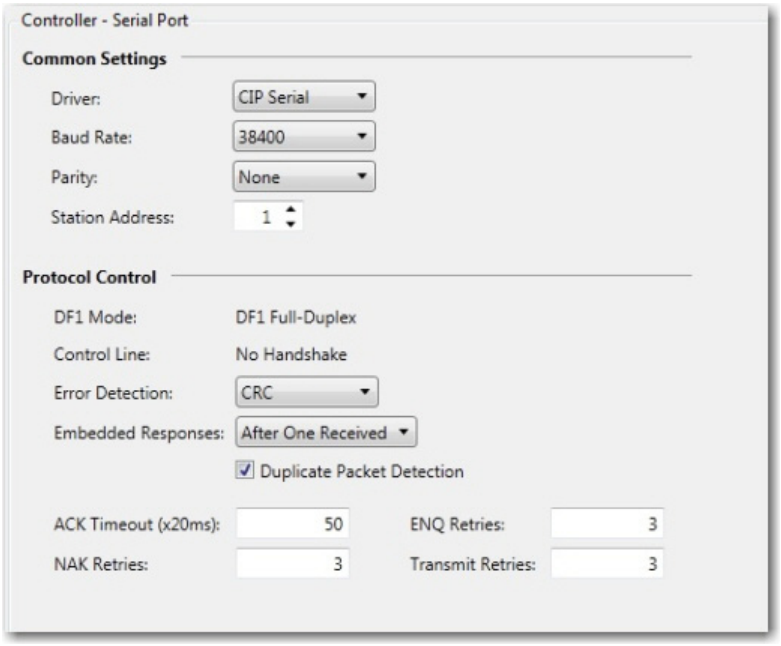
You can configure the Serial Port driver as CIP Serial, Modbus RTU, ASCII or Shutdown through the Device Configuration tree in Connected Components Workbench.

Configure CIP Serial Driver

1. Open your Connected Components Workbench project. On the device configuration tree, go to the Controller properties. Click Serial Port.



2. Select CIP Serial from the Driver field.



3. Specify a baud rate. Select a communication rate that all devices in your system support. Configure all devices in the system for the same communication rate. Default baud rate is set at 38400 bps.
4. In most cases, parity and station address should be left at default settings.

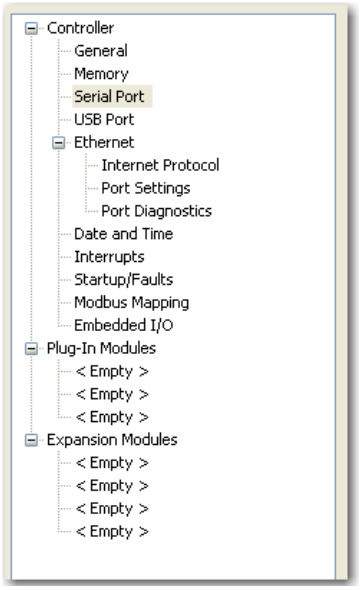
5. Click Advanced Settings and set Advanced parameters. Refer to the table CIP Serial Driver Parameters on page 60 for a description of the CIP Serial parameters.

CIP Serial Driver Parameters

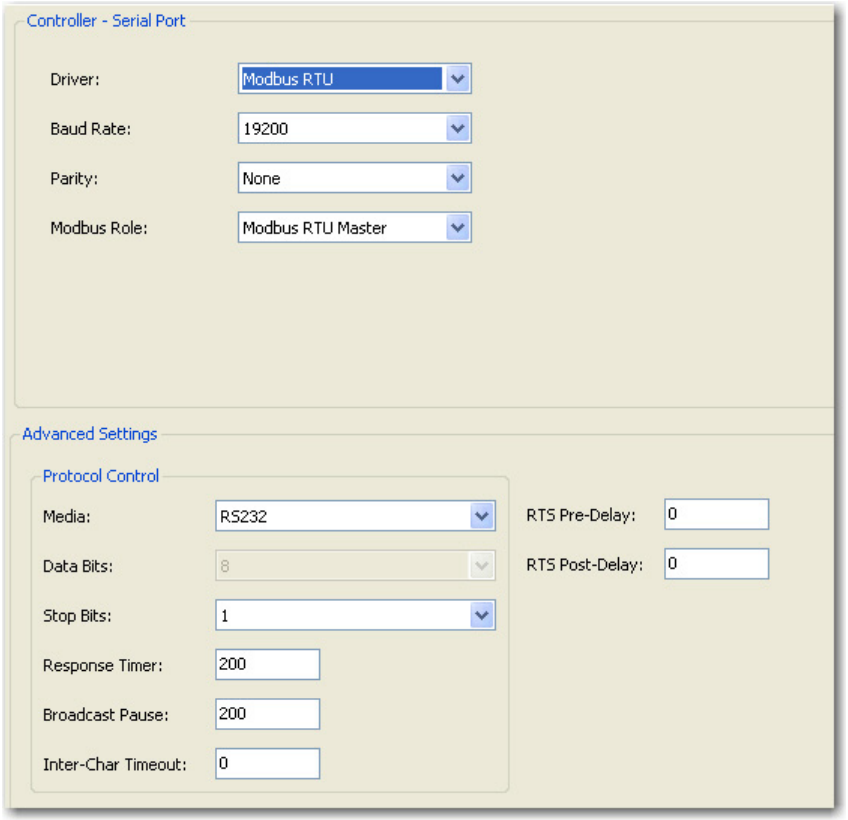
Parameter	Options	Default
Baud rate	Toggles between the communication rate of 1200, 2400, 4800, 9600, 19200, and 38400.	38400
Parity	Specifies the parity setting for the serial port. Parity provides additional message-packet error detection. Select Even, Odd, or None.	None
Station Address	The station address for the serial port on the DF1 master. The only valid address is 1.	1
DF1 Mode	DF1 Full Duplex (read only)	Configured as full-duplex by default.
Control Line	No Handshake (read only)	Configured as no handshake by default.
Duplicate Packet Detection	Detects and eliminates duplicate responses to a message. Duplicate packets may be sent under noisy communication conditions when the sender's retries are not set to 0. Toggles between Enabled and Disabled.	Enabled
Error Detection	Toggles between CRC and BCC.	CRC
Embedded Responses	To use embedded responses, choose Enabled Unconditionally. If you want the controller to use embedded responses only when it detects embedded responses from another device, choose After One Received. If you are communicating with another Allen-Bradley device, choose Enabled Unconditionally. Embedded responses increase network traffic efficiency.	After One Received
NAK Retries	The number of times the controller will resend a message packet because the processor received a NAK response to the previous message packet transmission.	3
ENQ Retries	The number of enquiries (ENQs) that you want the controller to send after an ACK timeout occurs.	3
Transmit Retries	Specifies the number of times a message is retried after the first attempt before being declared undeliverable. Enter a value from 0…127.	3
ACK Timeout (x20 ms)	Specifies the amount of time after a packet is transmitted that an ACK is expected.	50

Configure Modbus RTU

1. Open your Connected Components Workbench project. On the device configuration tree, go to the Controller properties. Click Serial Port.



2. Select Modbus RTU on the Driver field.



3. Specify the following parameters:
   - Baud rate
   - Parity
   - Unit address
   - Modbus Role (Master, Slave, Auto)

Modbus RTU Parameters

Parameter	Options	Default
Baud Rate	1200, 2400, 4800, 9600, 19200, 38400	19200
Parity	None, Odd, Even	None
Modbus Role	Master, Slave, Auto	Master

4. Click Advanced Settings to set advanced parameters.
   Refer to the table for available options and default configuration for advanced parameters.

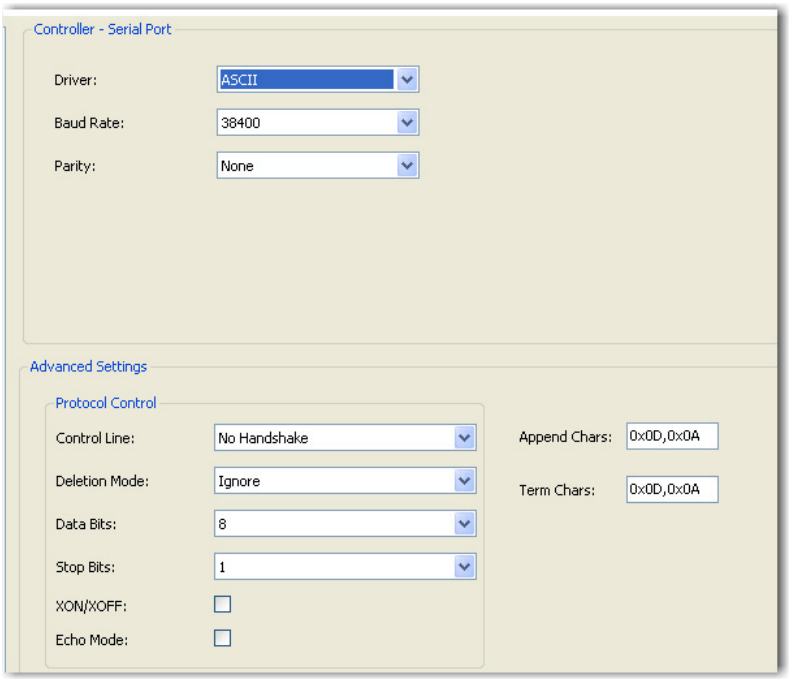
Modbus RTU Advanced Parameters

Parameter	Options	Default
Media	RS-232, RS-232 RTS/CTS, RS-485	RS-232
Data bits	Always 8	8
Stop bits	1, 2	1
Response timer	0…999,999,999 milliseconds	200
Broadcast Pause	0…999,999,999 milliseconds	200
Inter-char timeout	0…999,999,999 microseconds	0
RTS Pre-delay	0…999,999,999 microseconds	0
RTS Post-delay	0…999,999,999 microseconds	0

Configure ASCII

1. Open your Connected Components Workbench project. On the device configuration tree, go to Controller properties. Click Serial Port.

2. Select ASCII on the Driver field.

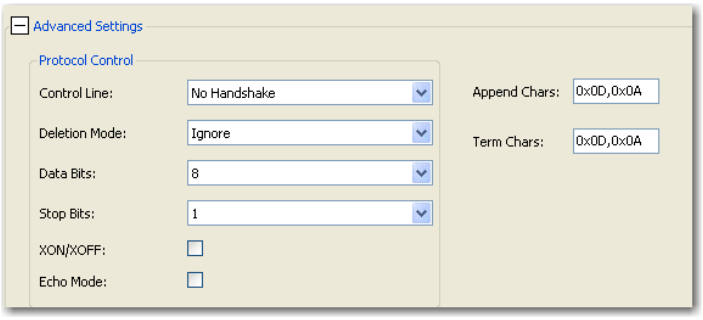


3. Specify baud rate and parity.

ASCII Parameters

Parameter	Options	Default
Baud Rate	1200, 2400, 4800, 9600, 19200, 38400	19200
Parity	None, Odd, Even	None

4. Click Advanced Settings to configure advanced parameters.

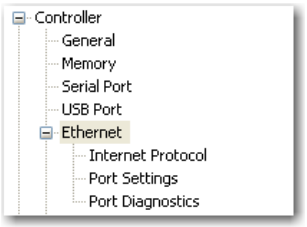


ASCII Advanced Parameters

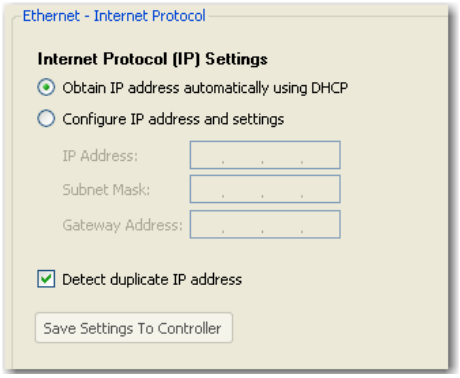
Parameter	Options	Default
Control Line	Full Duplex Half-duplex with continuous carrier Half-duplex without continuous carrier No Handshake	No Handshake
Deletion Mode	CRT Ignore Printer	Ignore
Data bits	7, 8	8
Stop bits	1, 2	1
XON/XOFF	Enabled or Disabled	Disabled
Echo Mode	Enabled or Disabled	Disabled
Append Chars	0x0D,0x0A or user-specified value	0x0D,0x0A
Term Chars	0x0D,0x0A or user-specified value	0x0D,0x0A

Configure Ethernet Settings

1. Open your Connected Components Workbench project (for example, Micro850). On the device configuration tree, go to Controller properties. Click Ethernet.



2. Under Ethernet, click Internet Protocol.
   Configure Internet Protocol (IP) settings. Specify whether to obtain the IP address automatically using DHCP or manually configure IP address, subnet mask, and gateway address.

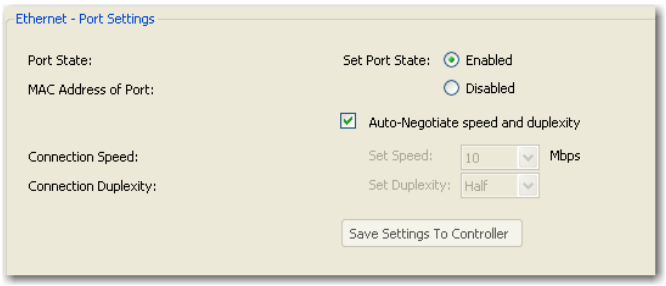


TIP The Ethernet port defaults to the following out-of-the box settings:

- DHCP (dynamic IP address)
- Address Duplicate Detection: On

IMPORTANT When a DHCP server fails, the Micro800 controller allocates IP addresses in the private range 169.254.0.1 to 169.254.255.254. The Micro800 controller verifies its address is unique on the network using ARP. When the DHCP server is again able to service requests, the Micro800 controller updates its address automatically.

3. Click the checkbox Detect duplicate IP address to enable detection of duplicate address.
4. Under Ethernet, click Port Settings.



5. Set Port State as Enabled or Disabled.
6. To manually set connection speed and duplexity, uncheck the option box Auto-Negotiate speed and duplexity. Then, set Speed (10 or 100 Mbps) and Duplexity (Half or Full) values.
7. Click Save Settings to Controller if you would like to save the settings to your controller.

8. On the device configuration tree, under Ethernet, click Port Diagnostics to monitor Interface and Media counters. The counters are available and updated when the controller is in Debug mode.

Validate IP Address

Modules must validate the incoming IP address configuration, whether it is obtained through explicit configuration or through DHCP.

The following rules must be obeyed when configuring the IP address:

- The IP address for the module cannot be set to zero, a multicast address, a broadcast address, or an address on the Class A loopback network (127.x.x.x).
- The IP address should not start with zero, and the IP address network ID should be not zero.
- The Network mask cannot be set to 255.255.255.255.
- The Gateway address must be on the same subnet as the IP address that is being configured.
- The Name Server address cannot be set to zero, a multicast address, a broadcast address, or an address on the Class A loopback network (127.x.x.x).

The valid range of static IPv4 IP address exclude:

- Broadcast or zero IP (255.255.255.255 or 0.0.0.0)
- IP address starting with 0 or 127 (0.xxx.xxx.xxx or 127.xxx.xxx.xxx)
- IP address ending with 0 or 255 (xxx.xxx.xxx.0 or xxx.xxx.xxx.255)
- IP addresses in range 169.254.xxx.xxx (169.254.0.0 to 169.254.255.255)
- IP addresses in range 224.0.0.0 to 255.255.255.255

Ethernet Host Name

Micro800 controllers implement unique host names for each controller, to be used to identify the controller on the network. The default host name is comprised of two parts: product type and MAC address, separated by a hyphen. For example: 2080LC50-xxxxxxxxxxxx, where xxxxxxxxxxxx is the MAC address.

The user can change the host name using the CIP Service Set Attribute Single when the controller is in Program/Remote Program mode.

Configure CIP Serial Driver

1. Open your Connected Components Workbench project. On the device configuration tree, go to the Controller properties. Click Serial port.
2. Select CIP Serial from the Driver field.
3. Specify a baud rate. Select a communication rate that all devices in your system support. Configure all devices in the system for the same communication rate. Default baud rate is set @ 38400 bps.
4. In most cases, parity and station address should be left at default settings.
5. Click Advanced Settings and set Advanced parameters.

OPC Support Using RSLinx Enterprise

Support for Open Platform Communications (OPC) using CIP symbolic has been added from firmware release 7.0 onwards. This can be used in place of Modbus addressing.

RSLinx Enterprise version 5.70 (CPR9 SR7) or later and FactoryTalk Gateway version 3.70 (CPR9 SR7) or later are required.

Notes:

Program Execution in Micro800

This section provides a brief overview of running or executing programs with a Micro800 controller.

IMPORTANT This section generally describes program execution in Micro800 controllers. Certain elements may not be applicable or true for certain models (for example, Micro820 does not support PTO motion control).

For detailed information regarding ladder diagrams, instructions, function blocks and so on, see the Micro800 Programmable Controllers Instruction Manual, publication 2080-RM001.

Overview of Program Execution

A Micro800 cycle or scan consists of reading inputs, executing programs in sequential order, updating outputs and performing housekeeping (data log, recipe, communications).

Program names must begin with a letter or underscore, followed by up to 127 letters, digits or single underscores. Use programming languages such as ladder logic, function block diagrams and structured text.

Up to 256 programs may be included in a project, depending on available controller memory. By default, the programs are cyclic (executed once per cycle or scan). As each new program is added to a project, it is assigned the next consecutive order number. When you start up the Project Organizer in Connected Components Workbench, it displays the program icons based on this order. You can view and modify an order number for a program from the program's properties. However, the Project Organizer does not show the new order until the next time the project is opened.

The Micro800 controller supports jumps within a program. Call a subroutine of code within a program by encapsulating that code as a User Defined Function (UDF) or User Defined Function Block (UDFB). A UDF is similar to a traditional subroutine and uses less memory than a UDFB, while a UDFB can have multiple instances. Although a UDFB can be executed within another UDFB, a maximum nesting depth of five is supported. A compilation error occurs if this is exceeded. This also applies to UDFs.

Alternatively, you can assign a program to an available interrupt and have it executed only when the interrupt is triggered. A program assigned to the User Fault Routine runs once just prior to the controller going into Fault mode.

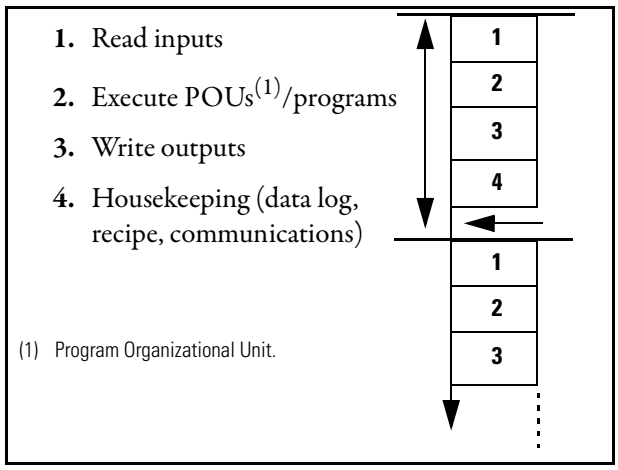
In addition to the User Fault Routine, Micro800 controllers also support two Selectable Timed Interrupts (STI). STIs execute assigned programs once every set point interval (1...65535 ms).

The Global System Variables associated with cycles/scans are:

- __SYSVA_CYCLECNT – Cycle counter
- __SYSVA_TCYCURRENT – Current cycle time
- __SYSVA_TCYMAXIMUM – Maximum cycle time since last start.

Execution Rules

This section illustrates the execution of a program. The execution follows four main steps within a loop. The loop duration is a cycle time for a program.



When a cycle time is specified, a resource waits until this time has elapsed before starting the execution of a new cycle. The POUs execution time varies depending on the number of active instructions. When a cycle exceeds the specified time, the loop continues to execute the cycle but sets an overrun flag. In such a case, the application no longer runs in real time.

When a cycle time is not specified, a resource performs all steps in the loop then restarts a new cycle without waiting.

Optional Module

Normally before the read inputs step, the controller will verify the presence of any configured plug-in and expansion I/O modules. If a plug-in or expansion I/O module is missing, the controller will fault. In Connected Components Workbench software release 10 or later, an Optional Module configuration option is added to prevent a missing plug-in I/O or expansion I/O module from faulting the controller if enabled. This option can be enabled separately for each plug-in I/O or expansion I/O module.

ATTENTION: If the optional module feature is enabled, use the MODULE_INFO instruction to verify that the module is present because the controller will not fault if the module is missing.

Controller Load and Performance Considerations

Within one program scan cycle, the execution of the main steps (as indicated in the Execution Rules diagram) could be interrupted by other controller activities which have higher priority than the main steps. Such activities include,

1. User Interrupt events, including STI, EII, and HSC interrupts (when applicable);
2. Communication data packet receiving and transmitting;
3. PTO Motion engine periodical execution (if supported by the controller).

When one or several of these activities occupy a significant percentage of the Micro800 controller execution time, the program scan cycle time will be prolonged. The Watchdog timeout fault (0xD011) could be reported if the impact of these activities is underestimated, and the Watchdog timeout is set marginally. The Watchdog setting defaults to 2 s and generally never needs to be changed.

Periodic Execution of Programs

For applications where periodic execution of programs with precise timing is required, such as for PID, it is recommended that STI (Selectable Timed Interrupt) be used to execute the program. STI provides precise time intervals.

It is not recommended that the system variable __SYSVA_TCYCYCTIME be used to periodically execute all programs as this also causes all communication to execute at this rate.

WARNING: Communication timeouts may occur if programmed cycle time is set too slow (for example, 200 ms) to maintain communications.

System Variable for Programmed Cycle Time

Variable	Type	Description
__SYSVA_TCYCYCTIME	TIME	Programmed cycle time. **Note**: Programmed cycle time only accepts values in multiples of 10 ms. If the entered value is not a multiple of 10, it will be rounded up to the next multiple of 10.

Power Up and First Scan

In Program mode, all analog and digital input variables hold their last state, and the LEDs are always updated. Also all analog and digital output variables hold their last state, but only the analog outputs hold their last state while the digital outputs are off.

When transitioning from Program mode to Run mode, all analog output variables hold their last state but all digital output variables are cleared.

Two system variables are also available from revision 2 and later.

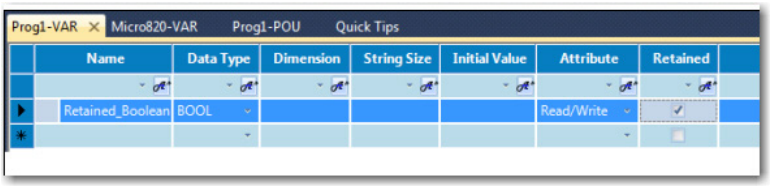
System Variables for Scan and Powerup on Firmware Release 2 and later

Variable	Type	Description
_SYSVA_FIRST_SCAN	BOOL	First scan bit. Can be used to initialize or reset variables immediately after every transition from Program to Run mode. **Note**: True only on first scan. After that, it is false.
_SYSVA_POWER_UP_BIT	BOOL	Powerup bit. Can be used to initialize or reset variables immediately after download from Connected Components Workbench or immediately after being loaded from memory backup module (for example, microSD card). **Note**: True only on the first scan after a powerup, or running a new ladder for the first time.

Variable Retention

Micro830 and Micro850 controllers retain all user-created variables after a power cycle, but the variables inside instances of instructions are cleared. For example: A user created variable called My_Timer of Time data type will be retained after a power cycle but the elapsed time (ET) within a user created timer TON instruction will be cleared.

Unlike Micro830/Micro850 controllers, Micro810 and Micro820 controllers can only retain a maximum of 400 bytes of user-created variable values. This means that after a power cycle, global variables are cleared or set to initial value, and only 400 bytes of user-created variable values are retained. Retained variables can be checked at the global variable page.



Memory Allocation

Depending on base size, available memory on Micro800 controllers are shown in the table below.

Memory Allocation for Micro800 Controllers

Attribute	10/16-point	20-point	24- and 48-points
Program steps(1)	4 K	10 K	10 K
Data bytes	8 KB	20 KB	20 KB

(1) Estimated Program and Data size are "typical" – program steps and variables are created dynamically. 1 Program Step = 12 data bytes.

These specifications for instruction and data size are typical numbers. When a project is created for Micro800, memory is dynamically allocated as either program or data memory at build time. This means that program size can exceed the published specifications if data size is sacrificed and vice versa. This flexibility allows maximum usage of execution memory. In addition to the user defined variables, data memory also includes any constants and temporary variables generated by the compiler at build time.

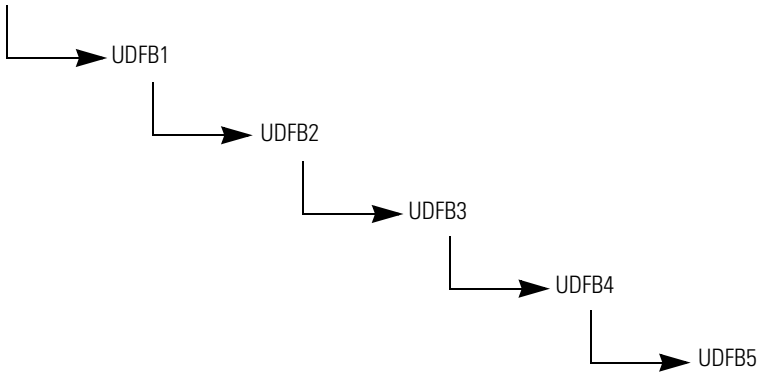
The Micro800 controllers also have project memory, which stores a copy of the entire downloaded project (including comments), as well as configuration memory for storing plug-in setup information, and so on.

Guidelines and Limitations for Advanced Users

Here are some guidelines and limitations to consider when programming a Micro800 controller using Connected Components Workbench software:

- Each program/POU can use up to 64 Kb of internal address space. It is recommended that you split large programs into smaller programs to improve code readability, simplify debugging and maintenance tasks.
- A User Defined Function (UDF) uses significantly less memory than a User Defined Function Block (UDFB). For example, 30% less for a typical sized program compared to a UDFB with one instance. The savings increases as the number of UDFB instances increases.
- A User Defined Function Block (UDFB) can be executed within another UDFB, with a limit of five nested UDFBs. Avoid creating UDFBs with references to other UDFBs, as executing these UDFBs too many times may result in a compile error. This also applies to UDFs.

Example of Five Nested UDFBs



- Structured Text (ST) is much more efficient and easier to use than Ladder Logic, when used for equations. If you are used to using the RSLogix 500 CPT Compute instruction, then a great alternative is to use ST combined with either UDF or UDFB.
  As an example, for an Astronomical Clock Calculation, Structured Text uses 40% less Instructions.

  Display_Output LD:
  Memory Usage (Code) : 3148 steps
  Memory Usage (Data) : 3456 bytes

  Display_Output ST:
  Memory Usage (Code) : 1824 steps
  Memory Usage (Data) : 3456 bytes
- You may encounter an Insufficient Reserved Memory error while downloading and compiling a program over a certain size. One workaround is to use arrays, especially if there are many variables.

Motion Control

Generally two types of motion control are used in Micro800 motion applications which have Kinetix® 3 servo drives.

- Indexed Motion – Micro800 issues position indexes to the servo drive using Modbus RTU communications or discrete I/O. Used for simple positioning. See publication CC-QS025 for building block example.
- PTO Motion – Micro800 controller uses pulse and direction outputs to the servo drive for precise control of position and velocity with Modbus RTU communications or discrete I/O for feedback. Micro800 motion configuration and instructions makes programming easy. See publication CC-QS033 for building block example.

PTO Motion Control

Certain Micro830 and Micro850 controllers (see table below) support motion control through high speed pulse-train outputs (PTO). PTO functionality refers to the ability of a controller to accurately generate a specific number of pulses at a specified frequency. These pulses are sent to a motion device, such as a servo drive, which in turn controls the number of rotations (position) of a servo motor. Each PTO is exactly mapped to one axis, to allow for control of simple positioning in stepper motors and servo drives with pulse/direction input.

As the duty cycle of the PTO can be changed dynamically, the PTO can also be used as a pulse width modulation (PWM) output.

PTO/PWM and motion axes support on the Micro830 and Micro850 controllers are summarized below.

PTO/PWM(1) and Motion Axis Support on Micro830 and Micro850

Controller	PTO (built-in)	Number of Axes Supported
10/16 Points(2) 2080-LC30-10QVB 2080-LC30-16QVB	1	1
24 Points 2080-LC30-24QVB(1) 2080-LC30-24QBB(1) 2080-LC50-24QVB 2080-LC50-24QBB	2	2
48 Points 2080-LC30-48QVB(1) 2080-LC30-48QBB(1) 2080-LC50-48QVB 2080-LC50-48QBB	3	3

(1) PWM outputs are only supported on firmware revision 6 and later.

(2) For Micro830 catalogs, Pulse Train Output functionality is only supported from firmware revision 2 and later.

ATTENTION: To use the Micro800 Motion feature effectively, users need to have a basic understanding of the following:

- PTO components and parameters
  See Use the Micro800 Motion Control Feature on page 76 for a general overview of Motion components and their relationships.
- Programming and working with elements in the Connected Components Workbench software
  The user needs to have a working knowledge of ladder diagram, structured text, or function block diagram programming to be able to work with motion function blocks, variables, and axis configuration parameters.

ATTENTION: To learn more about Connected Components Workbench and detailed descriptions of the variables for the Motion Function Blocks, you can refer to Connected Components Workbench Online Help that comes with your Connected Components Workbench installation.

IMPORTANT The PTO function can only be used with the controller's embedded I/O. It cannot be used with expansion I/O modules.

Use the Micro800 Motion Control Feature

The Micro800 motion control feature has the following elements. New users need to have a basic understanding of the function of each element to effectively use the feature.

Components of Motion Control

Element	Description	Page
Pulse Train Outputs	Consists of one pulse output and one direction output. A standard interface to control a servo or stepper drive.	• Input and Output Signals on page 78

Components of Motion Control

Axis	From a system point of view, an axis is a mechanical apparatus that is driven by a motor and drive combination. The drive receives position commands through the Micro800 pulse train outputs interface based upon the PLC execution of motion function blocks. On the Micro800 controller, it is a pulse train output and a set of inputs, outputs, and configuration.	• Motion Axis and Parameters on page 91 • Motion Axis Configuration in Connected Components Workbench on page 103
Motion Function Blocks	A set of instructions that configure or act upon an axis of motion.	• Connected Components Workbench Online Help • Motion Control Function Blocks on page 81 • Axis_Ref Data Type on page 98 • Function Block and Axis Status Error Codes on page 100 • Homing Function Block on page 116
Jerk	Rate of change of acceleration. The Jerk component is mainly of interest at the start and end of motion. Too high of a Jerk may induce vibrations.	• See Acceleration, Deceleration, and Jerk Inputs on page 83.

To use the Micro800 motion feature, you need to:

1. Configure the Axis Properties
   See Motion Axis Configuration in Connected Components Workbench on page 103 for instructions.
2. Write your motion program through the Connected Components Workbench software
   For instructions on how to use the Micro800 motion control feature, see the quickstart instructions, Use the Motion Control Feature on Micro800 Controllers, publication 2080-QS001.
3. Wire the Controller
   a. refer to Input and Output Signals on page 78 for fixed and configurable inputs/outputs
   b. See Sample Motion Wiring Configuration on 2080-LC30-xxQVB/2080-LC50-xxQVB on page 80 for reference

The next sections provide a more detailed description of the motion components. You can also refer to the Connected Components Workbench Online Help for more information about each motion function block and their variable inputs and outputs.

Input and Output Signals

Multiple input/output control signals are required for each motion axis, as described in the next tables. PTO Pulse and PTO Direction are required for an axis. The rest of the input/outputs can be disabled and re-used as regular I/O.

Fixed PTO Input/Output

Motion Signals	PTO0 (EM_00)		PTO1 (EM_01)		PTO2 (EM_02)	
	Logical Name in Software	Name on Terminal Block	Logical Name in Software	Name on Terminal Block	Logical Name in Software	Name on Terminal Block
PTO pulse	_IO_EM_DO_00	O-00	_IO_EM_DO_01	O-01	IO_EM_DO_02	O-02
PTO direction	_IO_EM_DO_03	O-03	_IO_EM_DO_04	O-04	IO_EM_DO_05	O-05
Lower (Negative) Limit switch	_IO_EM_DI_00	I-00	_IO_EM_DI_04	I-04	IO_EM_DI_08	I-08
Upper (Positive) Limit switch	_IO_EM_DI_01	I-01	_IO_EM_DI_05	I-05	IO_EM_DI_09	I-09
Absolute Home switch	_IO_EM_DI_02	I-02	_IO_EM_DI_06	I-06	IO_EM_DI_10	I-10
Touch Probe Input switch	_IO_EM_DI_03	I-03	_IO_EM_DI_07	I-07	IO_EM_DI_11	I-11

Configurable input/output

Motion Signals	Input/Output	Notes
Servo/Drive On	OUTPUT	Can be configured as any embedded output.
Servo/Drive Ready	INPUT	Can be configured as any embedded input.
In-Position signal (from Servo/motor)	INPUT	Can be configured as any embedded input.
Home Marker	INPUT	Can be configured as any embedded input, from input 0...15.

These I/O can be configured through the axis configuration feature in Connected Components Workbench. Any outputs assigned for motion should not be controlled in the user program.

See Motion Axis Configuration in Connected Components Workbench on page 103.

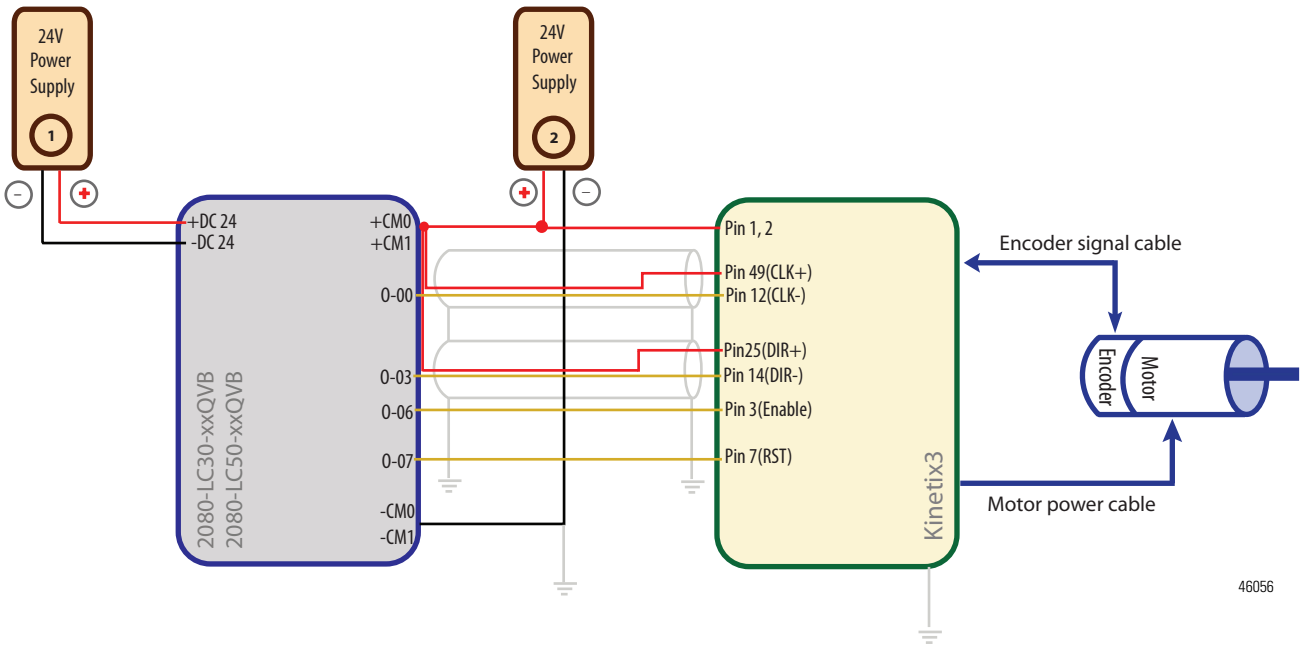
IMPORTANT If an output is configured for motion, then that output can no longer be controlled or monitored by the user program and cannot be forced. For example, when a PTO Pulse output is generating pulses, the corresponding logical variable IO_EM_DO_xx will not toggle its value and will not display the pulses in the Variable Monitor but the physical LED will give an indication.

If an input is configured for motion, then forcing the input only affects the user program logic and not motion. For example, if the input Drive Ready is false, then the user cannot force Drive Ready to true by forcing the corresponding logical variable IO_EM_DI_xx to be true.

Motion Wiring Input/Output Description

Motion Signals	Input/Output	Description	Uniqueness
PTO pulse	OUTPUT	PTO pulse from the embedded fast output, to be connected to Drive PTO input.	Not Shared
PTO direction	OUTPUT	PTO pulse direction indication, to be connected to Drive Direction input.	Not Shared
Servo/Drive On	OUTPUT	The control signal used to activate/deactivate Servo/Drive. This signal becomes Active when MC_Power (on) is commanded.	Can be shared with more than one drive
Lower (Negative) Limit switch	INPUT	The input for hardware negative limit switch, to be connected to mechanical/electrical negative limit sensor.	Not Shared
Upper (Positive) Limit switch	INPUT	The input for hardware positive limit switch, to be connected to mechanical/electrical positive limit sensor.	Not Shared
Absolute Home switch	INPUT	The input for hardware home switch (sensor), to be connected to mechanical/electrical home sensor.	Not Shared
Touch Probe Input switch	INPUT	The input for hardware touch probe signal, to be used with Motion MC_TouchProbe and MC_AbortTrigger function blocks to capture axis commanded position during the motion path.	Not Shared
Servo/Drive Ready	INPUT	The input signal that indicates Servo/Drive is ready to receive PTO pulse and direction signal from controller. No moving function blocks can be issued to an axis before the axis has this signal ready if this signal is Enabled in the motion axis configuration or axis properties page.	Can be shared with more than one drive
In-Position signal (from Servo/motor)	INPUT	The input signal that indicates the moving part is in the commanded position. This signal has to be Active after the moving part reaches the commanded position for MoveAbsolute and MoveRelative function blocks. For MoveAbsolute and MoveRelative function blocks, when In_Position is enabled, the controller will report an error (EP_MC_MECHAN_ERR) if the signal is not active within five seconds when the last PTO pulse sent out.	Not Shared
Home Marker	INPUT	This signal is the zero pulse signal from the motor encoder. This signal can be used for fine homing sequence to improve the homing accuracy.	Not Shared

Sample Motion Wiring Configuration on 2080-LC30-xxQVB/2080-LC50-xxQVB

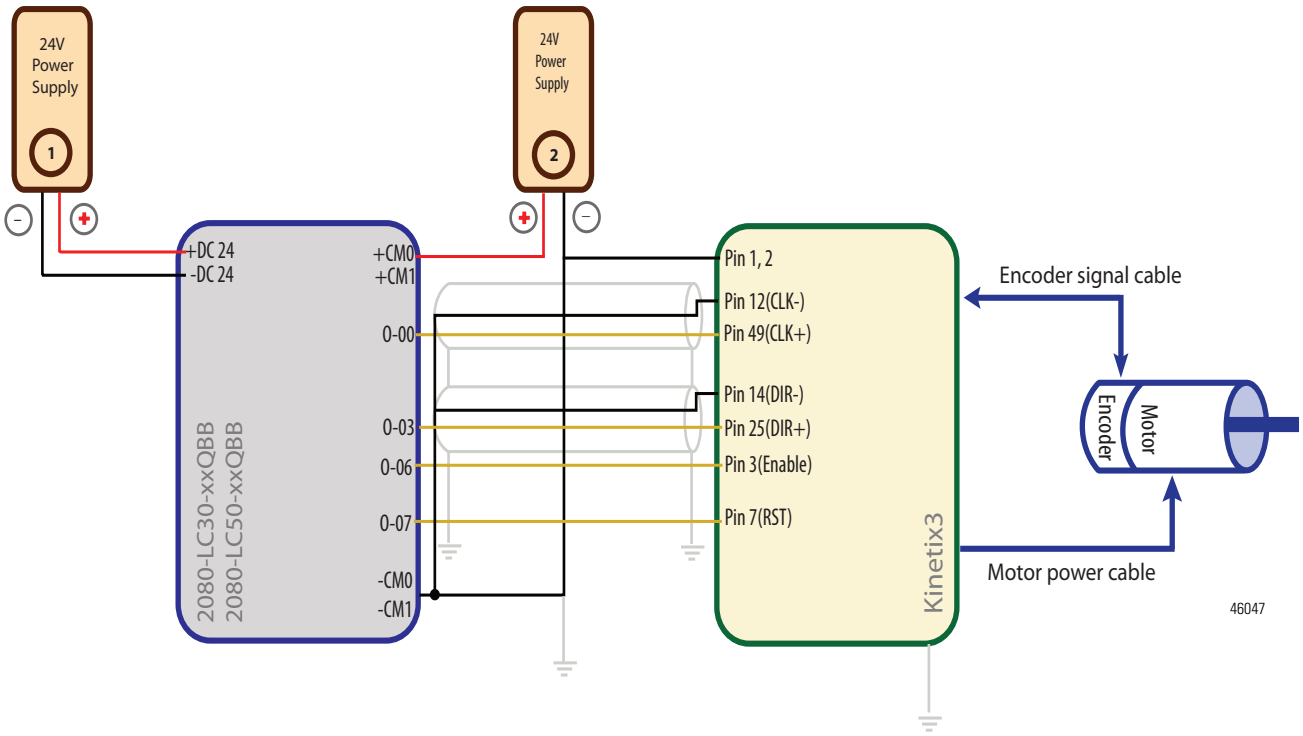


Notes:

(1) Drive Enable (Pin 3) and Reset Drive (Pin 7) will be operating as sourcing inputs when (Pin1,2) connected to ⊖ of the Power Supply 2.

(2) To help you configure Kinetix3 drive parameters so the drive can communicate and be controlled by a Micro830/Micro850 controller, see publication CC-QS033. The parameter Command Type must be set to "Step/Direction.Positive Logic", and the parameter Controller Output Type must be set to "Open Collector Input".

Sample Motion Wiring Configuration on 2080-LC30-xxQBB/2080-LC50-xxQBB



Notes:

(1) Drive Enable (Pin 3) and Reset Drive (Pin 7) will be operating as sinking inputs when (Pin 1,2) connected to ⊕ of the Power Supply 2.

(2) To help you configure Kinetix3 drive parameters so the drive can communicate and be controlled by a Micro830/Micro850 controller, see publication CC-QS033. The parameter Command Type must be set to "Step/Direction.Positive Logic", and the parameter Controller Output Type must be set to "Open Collector Input".

Motion Control Function Blocks

Motion control function blocks instruct an axis to a specified position, distance, velocity, and state.

Function Blocks are categorized as Movement (driving motion) and Administrative.

Administrative Function Blocks

Function Block Name
MC_Power
MC_Reset
MC_TouchProbe
MC_AbortTrigger
MC_ReadStatus
MC_SetPosition

Function Block Name
MC_ReadAxisError
MC_ReadParameter
MC_ReadBoolParameter
MC_WriteParameter
MC_WriteBoolParameter

WARNING: During Run Mode Change (RMC), the MC_Power function block should be disabled, which will power down the axis. Otherwise the axis will remain powered even if the function block is deleted.

Take note of the following:

- If a new instance of MC_Power accesses the axis, the axis will enter the error stop state.
- If MC_Power is inside a UDFB and any edit is made to the UDFB that changes the UDFB template (for example, adding a local variable), the axis will enter the error stop state.

Movement Function Blocks

Function Block Name	Description	Correct Axis State for issuing Function Block
MC_MoveAbsolute	This function block commands an axis to a specified absolute position.	Standstill, Discrete Motion, Continuous Motion
MC_MoveRelative	This function block commands an axis of a specified distance relative to the actual position at the time of execution.	Standstill, Discrete Motion, Continuous Motion
MC_MoveVelocity	This function block commands a never ending axis move at a specified velocity.	Standstill, Discrete Motion, Continuous Motion
MC_Home	This function block commands the axis to perform the “search home” sequence. The “Position” input is used to set the absolute position when reference signal is detected, and configured Home offset is reached. This function block completes at “StandStill” if the homing sequence is successful.	Standstill
MC_Stop	This function block commands an axis stop and transfers the axis to the state “Stopping”. It aborts any ongoing function block execution. While the axis is in state Stopping, no other function block can perform any motion on the same axis. After the axis has reached velocity zero, the Done output is set to TRUE immediately. The axis remains in the state “Stopping” as long as Execute is still TRUE or velocity zero is not yet reached. As soon as “Done” is SET and “Execute” is FALSE the axis goes to state “StandStill”.	Standstill, Discrete Motion, Continuous Motion, Homing
MC_Halt	This function block commands an axis to a controlled motion stop. The axis is moved to the state “DiscreteMotion”, until the velocity is zero. With the Done output set, the state is transferred to “StandStill”.	Standstill, Discrete Motion, Continuous Motion

ATTENTION: During Run Mode Change, the Movement Function Blocks can only be deleted when that Function Block has been done or aborted. Otherwise unintended axis and Function Block behavior may occur.

ATTENTION: Each motion function block has a set of variable inputs and outputs that allows you to control a specific motion instruction. Refer to the Connected Components Workbench Online Help for a description of these variable inputs and outputs.

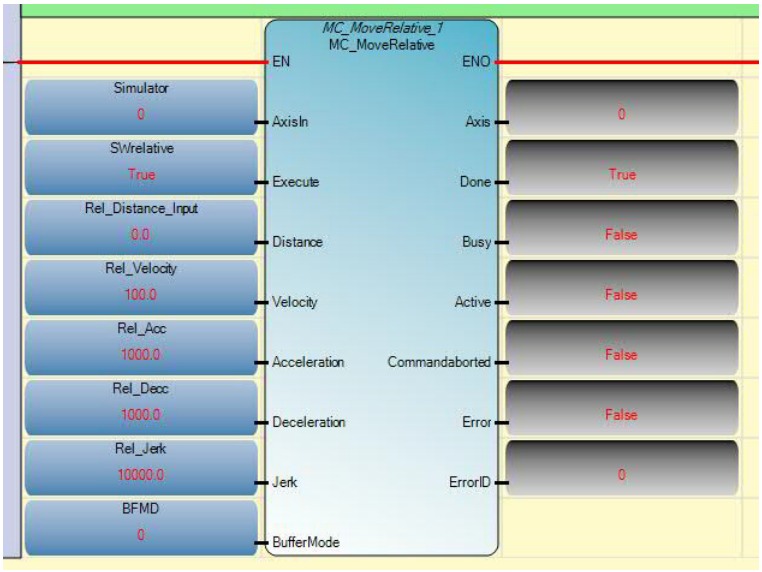
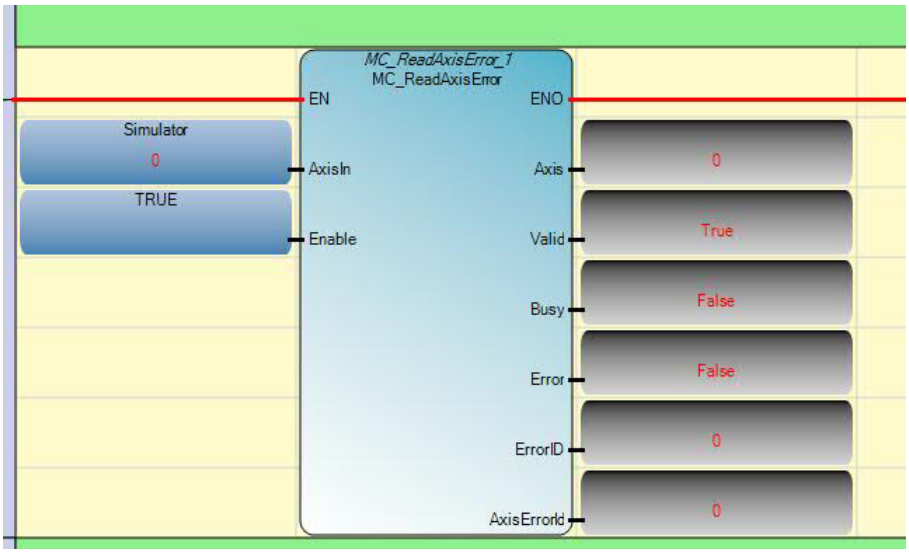
General Rules for the Motion Control Function Blocks

To work with motion control function blocks, users need to be familiar with the following general rules.

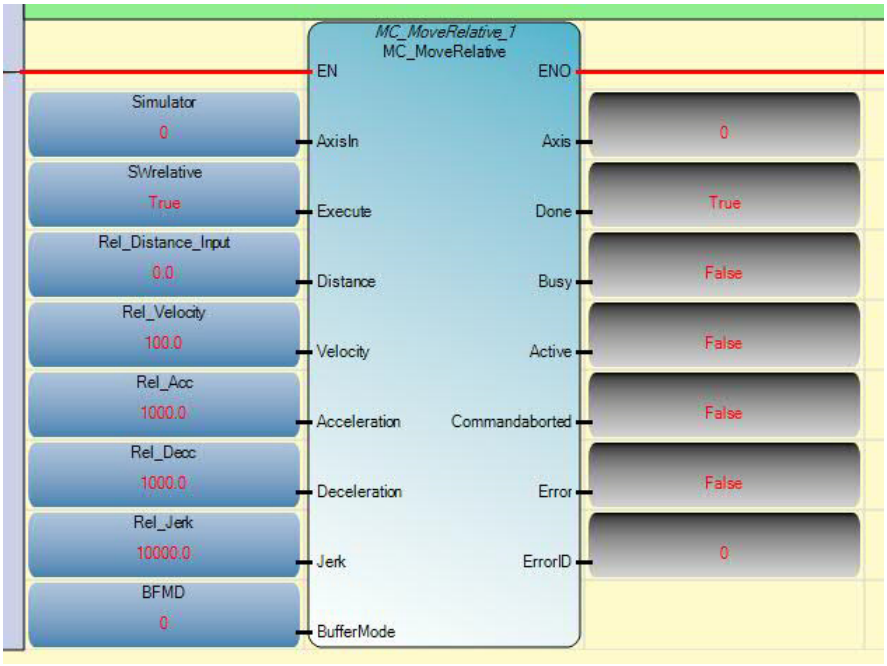
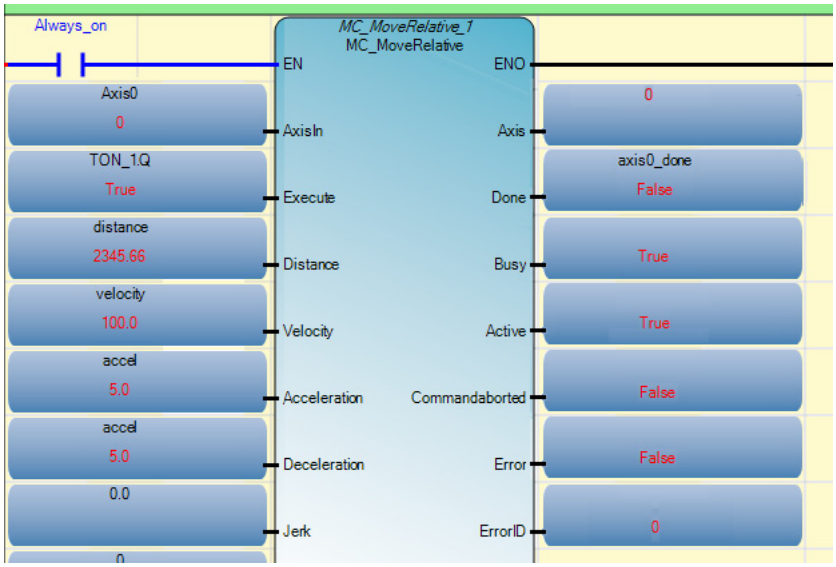
General Rules for the Motion Function Block

Parameter	General Rules
Input parameters	**When Execute is True**: The parameters are used with the rising edge of the Execute input. To modify any parameter, it is necessary to change the input parameter(s) and to trigger motion again. **When Enable is True**: The parameters are used with the rising edge of the Enable input and can be modified continuously.
Inputs exceeding application limits	If a function block is configured with parameters that result in a violation of application limits, the instance of the function block generates an error. The Error output will be flagged On, and error information will be indicated by the output ErrorID. The controller, in most cases, will remain in Run mode, and no motion error will be reported as a major controller fault.
Position/Distance Input	For MC_MoveAbsolute function block, the position input is the absolute location commanded to the axis. For MC_MoveRelative, the distance input is the relative location (considering current axis position is 0) from current position.
Velocity Input	Velocity can be a signed value. Users are advised to use positive velocity. Direction input for the MC_MoveVelocity function block can be used to define the direction of the move (that is, negative velocity x negative direction = positive velocity). For MC_MoveRelative and MC_MoveAbsolute function blocks the absolute value of the velocity is used. Velocity input does not need to be reached if Jerk input is equal to 0.
Direction Input	For MC_MoveAbsolute, direction input is ignored. (This is reserved for future use.) For MC_MoveVelocity, direction input value can be 1 (positive direction), 0 (current direction) or -1 (negative direction). For any other value, only the sign is taken into consideration. For example, -3 denotes negative direction, +2 denotes positive direction, and so on. For MC_MoveVelocity, the resulting sign of the product value derived from *velocity x direction* decides the motion direction, if the value is not 0. For example, if velocity x direction = +300, then direction is positive.
Acceleration, Deceleration, and Jerk Inputs	• Deceleration or Acceleration inputs should have a positive value. If Deceleration or Acceleration is set to be a non-positive value, an error will be reported (Error ID: MC_FB_ERR_RANGE). • The Jerk input should have a non-negative value. If Jerk is set to be a negative value, error will be reported. (Error ID: MC_FB_ERR_RANGE). • If maximum Jerk is configured as zero in Connected Components Workbench motion configuration, all jerk parameters for the motion function block has to be configured as zero. Otherwise, the function block reports an error (Error ID: MC_FB_ERR_RANGE). • If Jerk is set as a non-zero value, S-Curve profile is generated. If Jerk is set as zero, trapezoidal profile is generated. • If the motion engine fails to generate the motion profile prescribed by the dynamic input parameters, the function block reports an error (Error ID: MC_FB_ERR_PROFILE). See Function Block and Axis Status Error Codes on page 100 for more information about error codes.

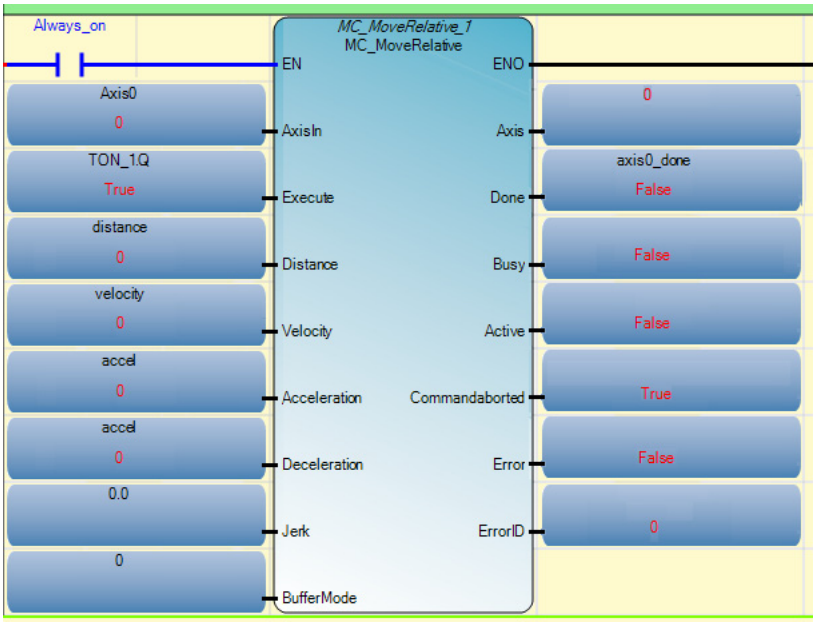
General Rules for the Motion Function Block

Parameter	General Rules
Output Exclusivity	**With Execute**: The outputs Busy, Done, Error, and CommandAborted indicate the state of the function block and are mutually exclusive – only one of them can be true on one function block. If execute is true, one of these outputs has to be true. The outputs Done, Busy, Error, ErrorID, and CommandAborted are reset with the falling edge of Execute. However, the falling edge of Execute does not stop or even influence the execution of the actual function block. Even if Execute is reset before the function block completes, the corresponding outputs are set for at least one cycle. If an instance of a function block receives a new Execute command before it completes (as a series of commands on the same instance), the new Execute command is ignored, and the previously issued instruction continues with execution.  **With Enable**: The outputs Valid and Error indicate whether a read function block executes successfully. They are mutually exclusive: only one of them can be true on one function block for MC_ReadBool, MC_ReadParameter, MC_ReadStatus. The Valid, Enabled, Busy, Error, and ErrorID outputs are reset with the falling edge of Enable as soon as possible. 
Axis output	When used in Function Block Diagram, you can connect the axis output parameter to the Axis input parameter of another motion function block for convenience (for example, MC_POWER to MC_HOME). When used in a Ladder Diagram, you cannot assign a variable to the Axis output parameter of another motion function block because it is read-only.

General Rules for the Motion Function Block

Parameter	General Rules
Behavior of Done Output	The output Done is set when the commanded action has completed successfully. With multiple function blocks working on the same axis in a sequence, the following rule applies: When one movement on an axis is aborted with another movement on the same axis without having reached the final goal, output Done will not be set on the first function block. 
Behavior of Busy Output	Every function block has a Busy output, indicating that the function block is not yet finished (for function blocks with an Execute input), and new output values are pending (for function blocks with Enable input). Busy is set at the rising edge of Execute and reset when one of the outputs Done, Aborted, or Error is set, or it is set at the rising edge of Enable and reset when one of the outputs Valid or Error is set. It is recommended that the function block continue executing in the program scan for as long as Busy is true, because the outputs will only be updated when the instruction is executing. For example, in ladder diagram, if the rung becomes false before the instruction finishes executing, the Busy output will stay true forever even though the function block has finished executing. 

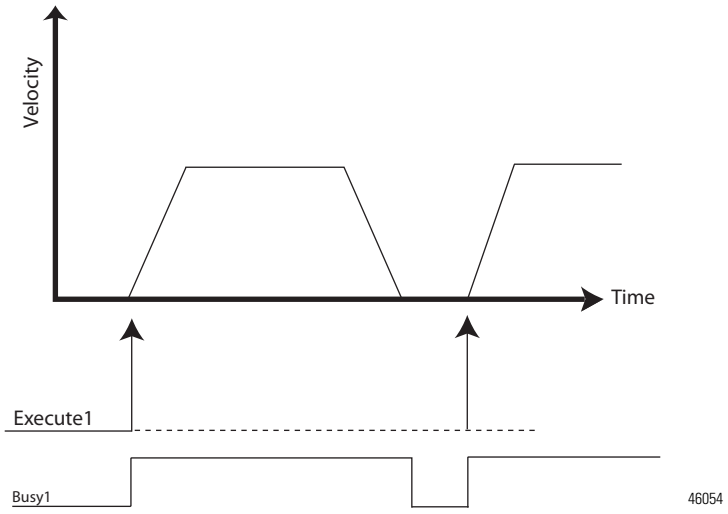
General Rules for the Motion Function Block

Parameter	General Rules
Output Active	In current implementation, buffered moves are not supported. Consequently, Busy and Active outputs have the same behavior.
Behavior of CommandAborted Output	CommandAborted is set when a commanded motion is aborted by another motion command. When CommandAborted occurs, other output signals such as InVelocity are reset. 
Enable and Valid Status	The Enable input for read function blocks is level-sensitive. On every program scan with the Enable input as true, the function block will perform a read and update its outputs. The Valid output parameter shows that a valid set of outputs is available. The Valid output is true as long as valid output values are available and the Enable input is true. The relevant output values will be refreshed as long as the input Enable is true. If there is a function block error, and the relevant output values are not valid, then the valid output is set to false. When the error condition no longer exists, the values will be updated and the Valid output will be set again.
Relative Move versus Absolute Move	Relative move does not require the axis to be homed. It simply refers to a move in a specified direction and distance. Absolute move requires that the axis be homed. It is a move to a known position within the coordinate system, regardless of distance and direction. Position can be negative or positive value.
Buffered Mode	For all motion control function blocks, BufferMode input parameter is ignored. Only aborted moves are supported for this release.
Error Handling	All blocks have two outputs which deal with errors that can occur during execution. These outputs are defined as follows: • **Error** – Rising edge of **"**Error**"** informs that an error occurred during the execution of the function block, where the function block cannot successfully complete. • **ErrorID** – Error number. **Types of errors**: • Function block logic (such as parameters out of range, state machine violation attempted) • hard limits or soft limits reached • Drive failure (Drive Ready is false) For more information about function block error, see Motion Function Block and Axis status Error ID on page 101.

Simultaneous Execution of Two Movement Function Blocks (Busy Output = True)

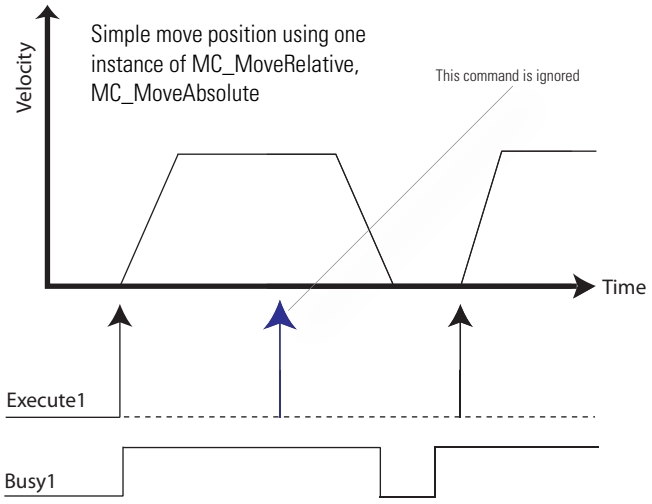
The general rule is that when a movement function block is busy, then a function block **with the same instance** (for example, MC_MoveRelative2) cannot be executed again until the function block status is not busy.

TIP MC_MoveRelative, MC_MoveAbsolute will be busy until final position is reached. MC_MoveVelocity, MC_Halt, and MC_Stop will be busy until final velocity is reached.



When a movement function block is busy, a function block **with a different instance** (for example, MC_MoveRelative1 and MC_MoveAbsolute1 on the same axis) can abort the currently executing function block. This is mostly useful for on-the-fly adjustments to position, velocity, or to halt after a specific distance.

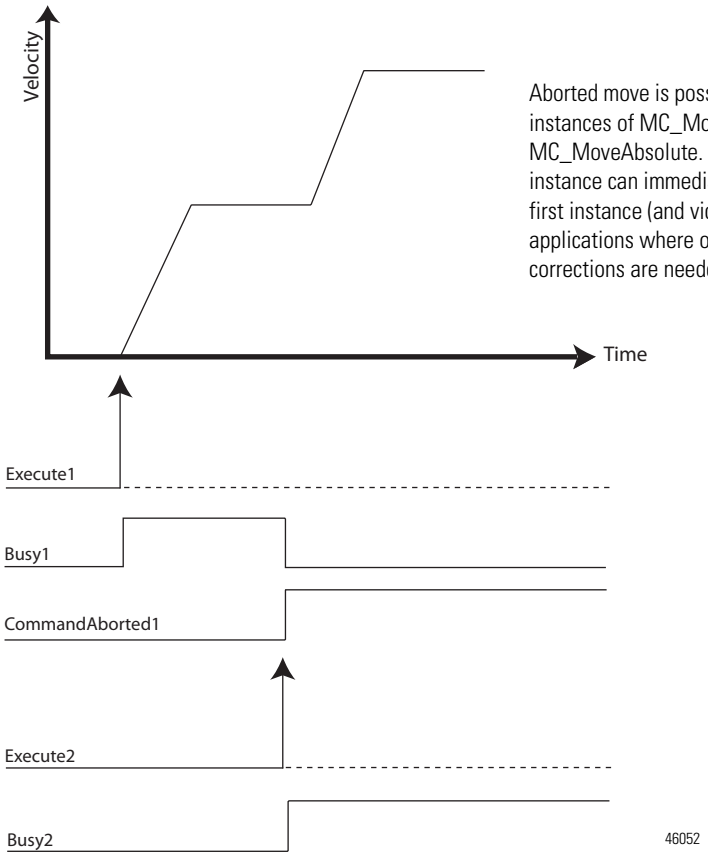
Example: Move to Position Ignored Due to Busy



For simple moves, the movement function block finishes. Busy output indicates that the function block is executing and must be allowed to finish before Execute input is toggled again.

If Execute is toggled again before Busy is false, the new command is ignored. No error is generated.

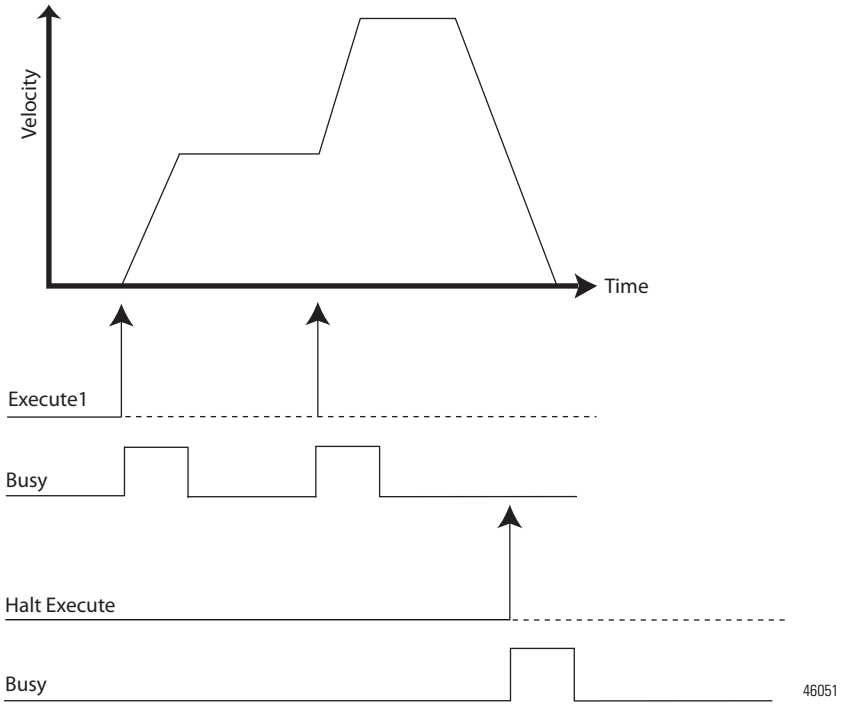
Example: Successful Aborted Move


Velocity
Aborted move is possible if using two instances of MC_MoveRelative, MC_MoveAbsolute. The second instance can immediately abort the first instance (and vice versa) for applications where on-the-fly corrections are needed.
Time
Execute1
Busy1
CommandAborted1
Execute2
Busy2
46052

Example: Changing Velocity With No Abort

When changing velocity, generally, an aborted move is not necessary since the function block is only Busy during acceleration (or deceleration). Only a single instance of the function block is required.

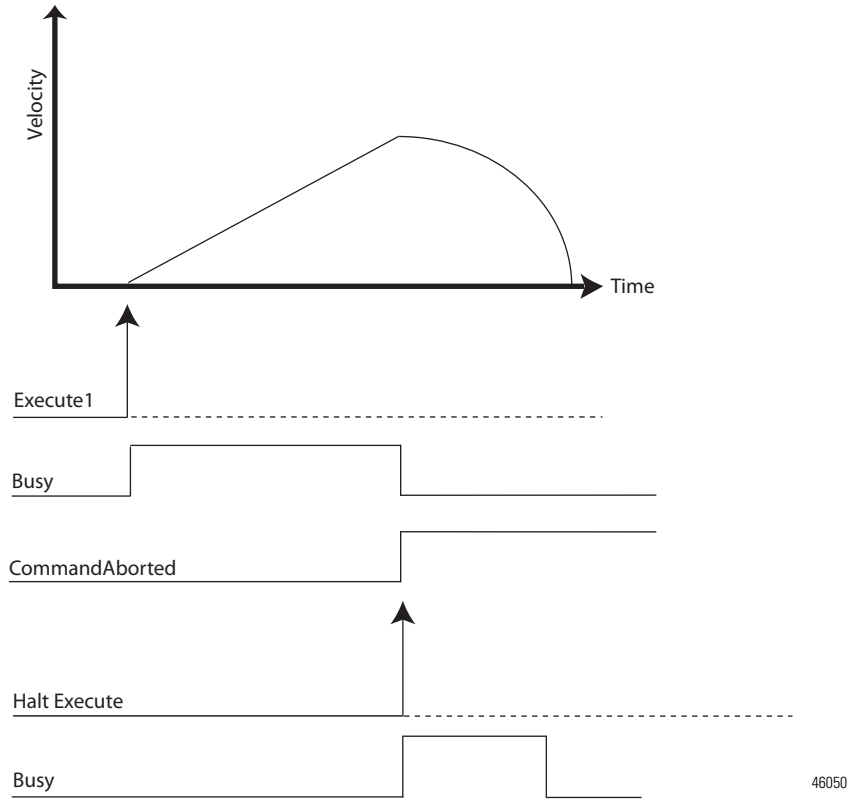
To bring the axis to a standstill, use MC_Halt.



It is possible for the movement function blocks and MC_Halt to abort another motion function block during acceleration/deceleration. This is not recommended as the resulting motion profile may not be consistent.

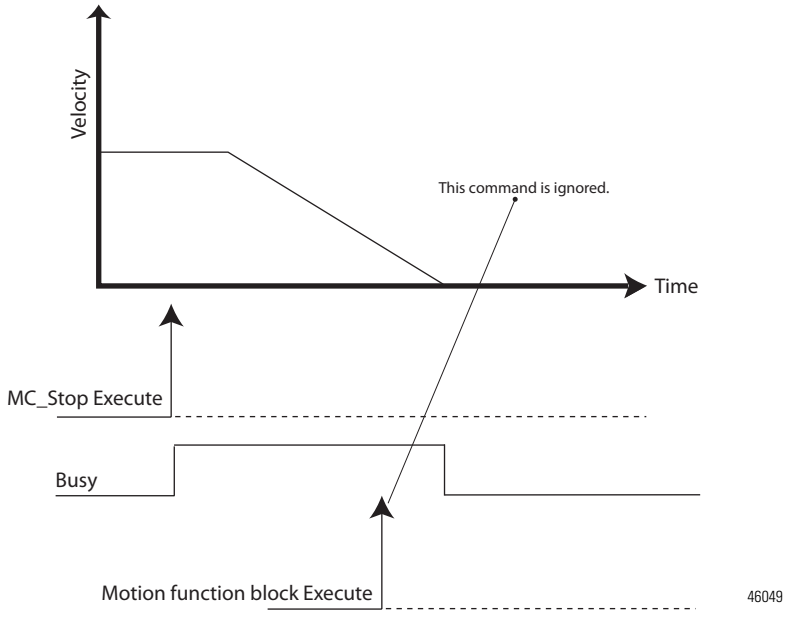
ATTENTION: If MC_Halt aborts another motion function block during acceleration and the MC_Halt Jerk input parameter is less than the Jerk of the currently executing function block, the Jerk of the currently executing function block is used to prevent an excessively long deceleration.

Example: Aborted Movement Function Block During Acceleration/Deceleration



IMPORTANT If MC_Halt aborts another movement function block during acceleration and the MC_Halt Jerk input parameter is less than the Jerk of the currently executing FB, the Jerk of the currently executing function block is used to prevent excessively long deceleration.

Example: Error Stop using MC_Stop cannot be Aborted



MC_Halt and MC_Stop are both used to bring an axis to a Standstill but MC_Stop is used when an abnormal situation occurs.

TIP MC_Stop can abort other motion function blocks but can never be aborted itself.

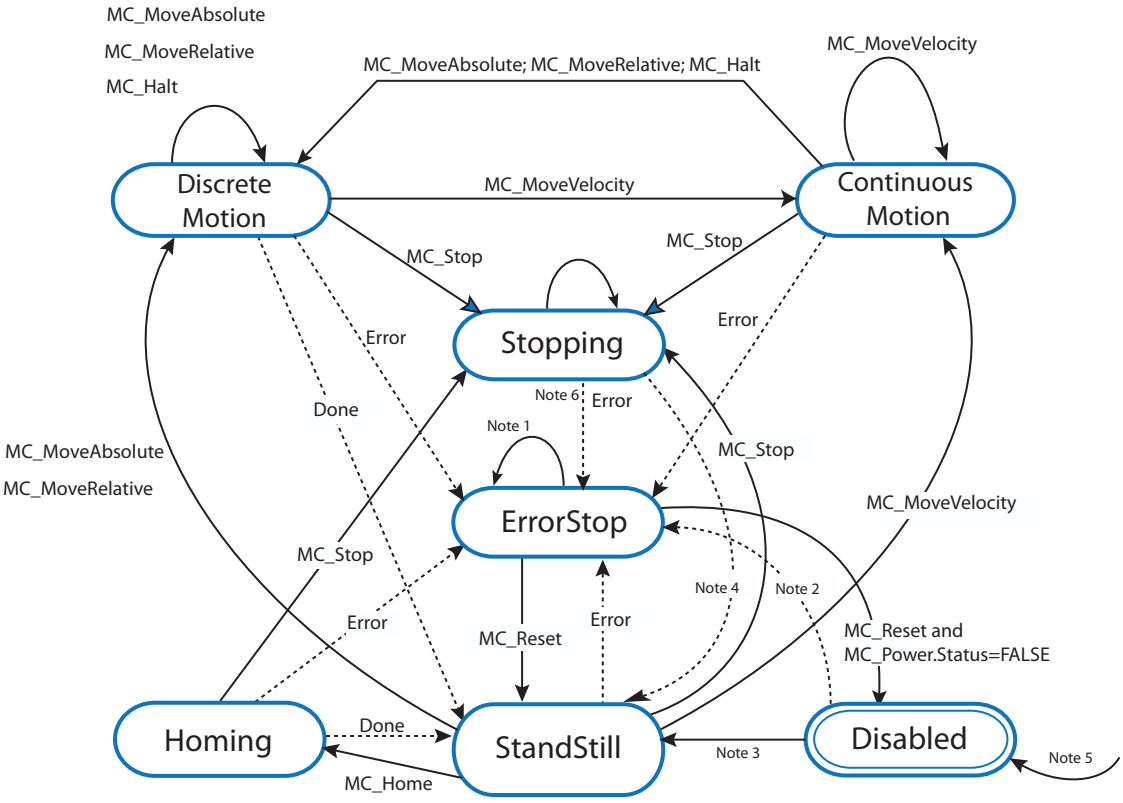
TIP MC_Stop goes to the Stopping state and normal operation cannot resume.

Motion Axis and Parameters

The following state diagram illustrates the behavior of the axis at a high level when multiple motion control function blocks are activated. The basic rule is that motion commands are always taken sequentially, even if the controller has the capability of real parallel processing. These commands act on the axis' state diagram.

The axis is always in one of the defined states (see diagram below). Any motion command is a transition that changes the state of the axis and, as a consequence, modifies the way the current motion is computed.

Motion Axis State Diagram



NOTES:

(1) In the ErrorStop and Stopping states, all function blocks (except MC_Reset), can be called although they will not be executed.
MC_Reset generates a transition to the Standstill state. If an error occurs while the state machine is in the Stopping state, a transition to the ErrorStop state is generated.
Axis position still updates even if the Axis state is ErrorStop. Also, the MC_TouchProbe function block is still active if it was executed before the ErrorStop state.

(2) Power.Enable = TRUE and there is an error in the Axis.

(3) Power.Enable = TRUE and there is no error in the Axis.

(4) MC_Stop.Done AND NOT MC_Stop.Execute.

(5) When MC_Power is called with Enable = False, the axis goes to the Disabled state for every state including ErrorStop.

(6) If an error occurs while the state machine is in Stopping state, a transition to the ErrorStop state is generated.

Axis States

The axis state can be determined from one of the following predefined states. Axis state can be monitored through the Axis Monitor feature of the Connected Components Workbench software when in debug mode.

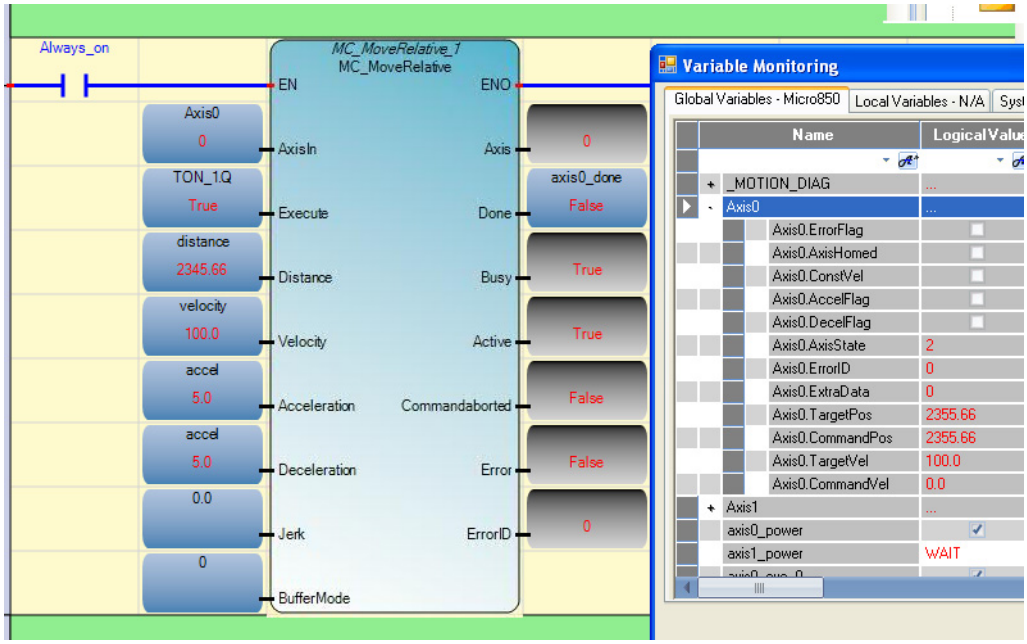
Motion States

State value	State Name
0x00	Disabled
0x01	Standstill
0x02	Discrete Motion
0x03	Continuous Motion
0x04	Homing
0x06	Stopping
0x07	Stop Error

Axis State Update

On motion execution, although the motion profile is controlled by Motion Engine as a background task, which is independent from POU scan, axis state update is still dependent on when the relevant motion function block is called by the POU scan.

For example, on a moving axis on a Ladder POU (state of a rung=true), an MC_MoveRelative function block in the rung is scanned and the axis starts to move. Before MC_MoveRelative completes, the state of the rung becomes False, and MC_MoveRelative is no longer scanned. In this case, the state of this axis cannot switch from Discrete Motion to StandStill, even after the axis fully stops, and the velocity comes to 0.



Limits

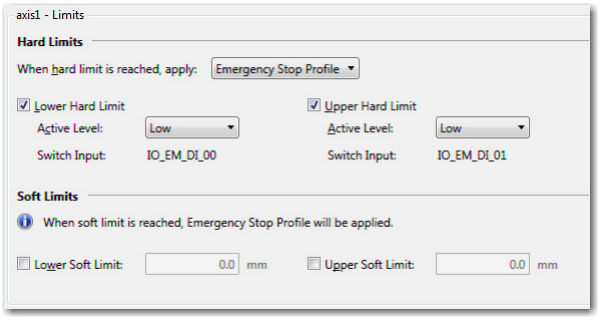
The Limits parameter sets a boundary point for the axis, and works in conjunction with the Stop parameter to define a boundary condition for the axis on the type of stop to apply when certain configured limits are reached.

There are three types of motion position limits.

- Hard Limits
- Soft Limits
- PTO Pulse Limits

TIP See Motion Axis Configuration in Connected Components Workbench on page 103 for information on how to configure limits and stop profiles and the acceptable value range for each.

If any one of these limits is reached on a moving axis (except on homing), an over travel limit error will be reported and the axis will be stopped based on configured behavior.



Sample Limits configuration in Connected Components Workbench

Hard Limits

Hard limits refer to the input signals received from physical hardware devices such as limit switches and proximity sensors. These input signals detect the presence of the load at the maximum upper and minimum lower extents of allowable motion of the load or movable structure that carries the load, such as a load tray on a transfer shuttle.

Hardware limits are mapped to discrete inputs that are associated with data tags/variables.

When a hard limit switch is enabled, the axis comes to a stop when the limit switch is detected during motion. If hard stop on hard limit switch is configured as ON and the limit is detected, motion is stopped immediately (that is, PTO pulse is stopped immediately by the hardware). Alternatively, if hard stop on hard limit switch is configured as OFF, motion will be stopped using Emergency Stop parameters.

When any hard limit switch is enabled, the input variable connecting to this physical input can still be used in User Application.

When a hard limit switch is enabled, it will be used automatically for MC_Home function block, if the switch is in the Homing direction configured in the Connected Components Workbench software (Mode: MC_HOME_ABS_SWITCH or MC_HOME_REF_WITH_ABS). See Homing Function Block on page 116.

Soft Limits

Soft limits refer to data values that are managed by the motion controller. Unlike hardware limits which detect the presence of the physical load at specific points in the allowable motion of the load, soft limits are based on the stepper commands and the motor and load parameters.

Soft limits are displayed in user defined units. The user can enable individual soft limits. For non-enabled soft limits (whether upper or lower), an infinite value is assumed.

Soft Limits are activated only when the corresponding axis is homed. Users can enable or disable soft limits, and configure an upper and lower limit setting through the Connected Components Workbench software.

Soft Limits Checking on the Function Blocks

Function Block	Limits Checking
MC_MoveAbsolute	The target position will be checked against the soft limits before motion starts.
MC_MoveRelative	
MC_MoveVelocity	The soft limits will be checked dynamically during motion.

When a soft limit is enabled, the axis comes to a stop when the limit is detected during motion. The motion is stopped using emergency stop parameters.

If both hard and soft limits are configured as enabled, for two limits in the same direction (upper or lower), the limits should be configured such that the soft limit is triggered before the hard limit.

PTO Pulse Limits

This limit parameter is not configurable by the user and is the physical limitation of the embedded PTO. The limits are set at 0x7FFF0000 and -0x7FFF0000 pulses, for upper and lower limits, respectively.

PTO pulse limits are checked by the controller unconditionally — that is, the checking is always ON.

On a non-continuous motion, to prevent a moving axis going to ErrorStop status with Motion PTO Pulse limits detected, user needs to prevent current position value going beyond PTO Pulse limit.

On a continuous motion (driven by MC_MoveVelocity function block), when the current position value goes beyond PTO pulse limit, PTO pulse current position will automatically roll over to 0 (or the opposite soft limit, if it is activated), and the continuous motion continues.

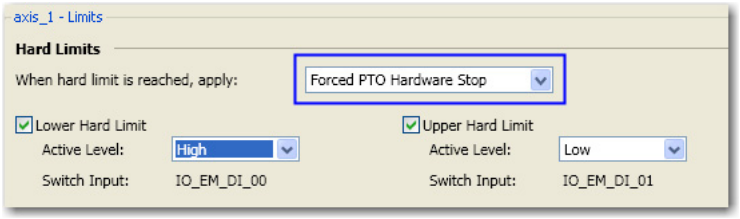
For a continuous motion, if the axis is homed, and the soft limit in the motion direction is enabled, soft limit will be detected before PTO pulse limit being detected.

Motion Stop

There are three types of stops that can be configured for an axis.

Immediate Hardware Stop

This type of Immediate Stop is controlled by the hardware. If a Hard Stop on a Hard Limit switch is enabled, and the Hard Limit has been reached, the PTO pulse for the axis will be cut off immediately by the controller. The stop response has no delay (less than 1 μs).



Immediate Soft Stop

The maximum possible response delay for this type of stop could be as much as the Motion Engine Execution time interval. This type of stop is applicable in the following scenarios:

- During motion, when axis PTO Pulse Limit is reached;
- One Hard Limit is enabled for an axis, but Hard Stop on Hard Limit switch is configured as Off. If the Emergency Stop is configured as Immediate Software Stop, during motion, when the Hard Limit switch is detected;
- One Soft Limit is enabled for an axis and the axis has been homed. If the emergency stop is configured as Immediate Soft Stop, during motion, when the Soft Limit reach is detected;

- The Emergency Stop is configured as Immediate Soft Stop. During motion, MC_Stop function block is issued with Deceleration parameter equal to 0.

Decelerating Soft Stop

Decelerating soft stop could be delayed as much as Motion Engine Execution Time interval. This type of stop is applied in the following scenarios:

- One Hard Limit is enabled for an axis, but Hard Stop on Hard Limit switch is configured as Off. If the emergency stop is configured as decelerating stop, during motion, when the Hard Limit switch is detected;
- One Soft Limit is enabled for an axis and the axis has been homed. If the emergency stop is configured as decelerating stop, during motion, when the soft limit reach is detected by firmware;
- The Emergency Stop is configured as Decelerating Stop. During motion, the MC_Stop function block is issued with deceleration parameter set to 0.
- During motion, MC_Stop function block is issued with Deceleration parameter not set to 0.

Motion Direction

For distance (position) motion, with the target position defined (absolute or relative), the direction input is ignored.

For velocity motion, direction input value can be positive (1), current (0) or negative (-1). For any other value, only the sign (whether positive or negative) is considered and defines whether the direction is positive or negative. This means that if the product of velocity and direction is -3, then direction type is negative.

MC_MoveVelocity Supported Direction Types

Direction Type	Value used[(1)]	Direction description
Positive direction	1	Specific for motion/rotation direction. Also called clockwise direction for rotation motion.
Current direction	0	Current direction instructs the axis to continue its motion with new input parameters, without direction change. The direction type is valid only when the axis is moving and the MC_MoveVelocity is called.
Negative direction	-1	Specific for motion/rotation direction. Also referred to as counter-clockwise direction for rotation motion.

(1) Data type: short integer.

Axis Elements and Data Types

Axis_Ref Data Type

Axis_Ref is a data structure that contains information on a motion axis. It is used as an input and output variable in all motion function blocks. One axis_ref instance is created automatically in the Connected Components Workbench software when the user adds one motion axis to the configuration.

The user can monitor this variable in controller debug mode through the software when the motion engine is active, or in the user application as part of user logic. It can also be monitored remotely through various communication channels.

Data Elements for Axis_Ref

Element name	Data Type	Description
Axis_ID	UINT8	The logic axis ID automatically assigned by the Connected Components Workbench software. This parameter cannot be edited or viewed by user.
ErrorFlag	UINT8	Indicates whether an error is present in the axis.
AxisHomed	UINT8	Indicates whether homing operation is successfully executed for the axis or not. When the user tries to redo homing for an axis with AxisHomed already set (homing performed successfully), and the result is not successful, the AxisHomed status will be cleared.
ConsVelFlag	UINT8	Indicates whether the axis is in constant velocity movement or not. Stationary axis is not considered to be in constant velocity.
AccFlag	UINT8	Indicates whether the axis is in an accelerating movement or not.
DecFlag	UINT8	Indicates whether the axis is in a decelerating movement or not.
AxisState	UINT8	Indicates the current state of the axis. For more information, see Axis States on page 93.
ErrorID	UINT16	Indicates the cause for axis error when error is indicated by ErrorFlag. This error usually results from motion function block execution failure. See Motion Function Block and Axis status Error ID on page 101.
ExtraData	UINT16	Reserved.
TargetPos	REAL (float)(1)	Indicates the final target position of the axis for MoveAbsolute and MoveRelative function blocks. For MoveVelocity, Stop, and Halt function blocks, TargetPos is 0 except when the TargetPos set by previous position function blocks is not cleared.

Data Elements for Axis_Ref

Element name	Data Type	Description
CommandPos	REAL (float)[1]	On a moving axis, this is the current position the controller commands the axis to go to.
TargetVel	REAL (float)[1]	The maximum target velocity issued to the axis by a move function block. The value of TargetVel is same as the velocity setting in current function block, or smaller, depending on other parameters in the same function block. This element is a signed value indicating direction information. See PTO Pulse Accuracy on page 114 for more information.
CommandVel	REAL (float)[1]	During motion, this element refers to the velocity the controller commands the axis to use. This element is a signed value indicating direction information.

(1) See Real Data Resolution on page 112 for more information on REAL data conversion and rounding.

IMPORTANT Once an axis is flagged with error, and the error ID is not zero, the user needs to reset the axis (using MC_Reset) before issuing any other movement function block.

IMPORTANT The update for axis status is performed at the end of one program scan cycle, and the update is aligned with the update of Motion Axis status.

Axis Error Scenarios

In most cases, when a movement function block instruction issued to an axis results in a function block error, the axis is also usually flagged as being in Error state. The corresponding ErrorID element is set on the axis_ref data for the axis. However, there are exception scenarios where an axis error is not flagged. The exception can be, but not limited to, the following scenarios:

- A movement function block instructs an axis, but the axis is in a state where the function block could not be executed properly. For example, the axis has no power, or is in Homing sequence, or in Error Stop state.
- A movement function block instructs an axis, but the axis is still controlled by another movement function block. The axis cannot allow the motion to be controlled by the new function block without going to a full stop. For example, the new function block commands the axis to change motion direction.
- When one movement function block tries to control an axis, but the axis is still controlled by another movement function block, and the newly-defined motion profile cannot be realized by the controller. For example, User Application issues an S-Curve MC_MoveAbsolute function block to an axis with too short a distance given when the axis is moving.
- When one movement function block is issued to an axis, and the axis is in the Stopping or Error Stopping sequence.

For the above exceptions, it is still possible for the user application to issue a successful movement function block to the axis after the axis state changes.

MC_Engine_Diag Data Type

The MC_Engine_Diag data type contains diagnostic information on the embedded motion engine. It can be monitored in debug mode through the Connected Components Workbench software when the motion engine is active, or through the user application as part of user logic. It can also be monitored remotely through various communication channels.

One MC_Engine_Diag instance is created automatically in the Connected Components Workbench software when the user adds the first motion axis in the motion configuration. This instance is shared by all user-configured motion axes.

Data Elements for MC_Engine_Diag

Element name	Data Type
MCEngState	UINT16
CurrScantime[1]	UINT16
MaxScantime[1]	UINT16
CurrEngineInterval[1]	UINT16
MaxEngineInterval[1]	UINT16
ExtraData	UINT16

(1) The time unit for this element is microsecond. This diagnostic information can be used to optimize motion configuration and user application logic adjustment.

MCEngstate States

State name	State	Description
MCEng_Idle	0x01	MC engine exists (at least one axis defined), but the engine is idle as there is no axis is moving. The Engine diagnostic data is not being updated.
MCEng_Running	0x02	MC engine exists (at least one axis defined) and the engine is running. The diagnostic data is being updated.
MCEng_Faulted	0x03	MC engine exists, but the engine is faulted.

Function Block and Axis Status Error Codes

All motion control function blocks share the same ErrorID definition.

Axis error and function block error share the same Error ID, but error descriptions are different, as described in the table below.

TIP Error code 128 is warning information to indicate the motion profile has been changed and velocity has been adjusted to a lower value but the function block can execute successfully.

Motion Function Block and Axis status Error ID

Error ID	Error ID MACRO	Error description for Function Block	Error description for Axis Status[1]
00	MC_FB_ERR_NO	Function block execution is successful.	The axis is in operational state.
01	MC_FB_ERR_WRONG_STATE	The function block cannot execute because the axis is not in the correct state. Check the axis state.	The axis is not operational due to incorrect axis state detected during a function block execution. Reset the state of the axis using the MC_Reset function block.
02	MC_FB_ERR_RANGE	The function block cannot execute because there is invalid axis dynamic parameter(s) (velocity, acceleration, deceleration, or jerk) set in the function block. Correct the setting for the dynamic parameters in the function block against Axis Dynamics configuration page.	The axis is not operational due to invalid axis dynamic parameter(s) (velocity, acceleration, deceleration, or jerk) set in a function block. Reset the state of the axis using the MC_Reset function block. Correct the setting for the dynamic parameters in the function block against Axis Dynamics configuration page.
03	MC_FB_ERR_PARAM	The function block cannot execute because there is invalid parameter other than velocity, acceleration, deceleration, or jerk, set in the function block. Correct the setting for the parameters (for example, mode or position) for the function block.	The axis is not operational due to invalid parameter(s) other than velocity, acceleration, deceleration, or jerk, set in a function block. Reset the state of the axis using the MC_Reset function block. Correct the setting for the parameters (for example, mode or position) for the function block.
04	MC_FB_ERR_AXISNUM	The function block cannot execute because the axis does not exist, the axis configuration data is corrupted, or the axis is not correctly configured.	Motion internal Fault, Error ID = 0x04. Call Tech support.
05	MC_FB_ERR_MECHAN	The function block cannot execute because the axis is faulty due to drive or mechanical issues. Check the connection between the drive and the controller (Drive Ready and In-Position signals), and ensure the drive is operating normally.	The axis is not operational due to drive or mechanical issues. Check the connection between the drive and the controller (Drive Ready and In-Position signals), and ensure the drive is operating normally. Reset the state of the axis using the MC_Reset function block.
06	MC_FB_ERR_NOPOWER	The function block cannot execute because the axis is not powered on. Power on the axis using MC_Power function block.	The axis is not powered on. Power on the axis using MC_Power function block. Reset the state of the axis using the MC_Reset function block.
07	MC_FB_ERR_RESOURCE	The function block cannot execute because the resource required by the function block is controlled by some other function block or not available. Ensure the resource required by the function block available for use. Some examples: • MC_power function block attempts to control the same axis. • MC_Stop function block is executed against the same axis at the same time. • Two or more MC_TouchProbe function blocks are executed against the same axis at the same time.	The axis is not operational due to the resource required by a function block is under the control of other function block, or not available. Ensure the resource required by the function block available for use. Reset the state of the axis using the MC_Reset function block.
08	MC_FB_ERR_PROFILE	The function block cannot execute because the motion profile defined in the function block cannot be achieved. Correct the profile in the function block.	The axis is not operational due to motion profile defined in a function block cannot be achieved. Reset the state of the axis using the MC_Reset function block. Correct the profile in the function block.

Motion Function Block and Axis status Error ID

Error ID	Error ID MACRO	Error description for Function Block	Error description for Axis Status[1]
09	MC_FB_ERR_VELOCITY	The function block cannot execute because the motion profile requested in the function block cannot be achieved due to current axis velocity. Some examples: • The function block requests the axis to reverse the direction while the axis is moving. • The required motion profile cannot be achieved due to current velocity too low or too high. Check the motion profile setting in the function block, and correct the profile, or re-execute the function block when the axis velocity is compatible with the requested motion profile.	The axis is not operational. The motion profile requested in the function block cannot be achieved because of current axis velocity. Some examples: • The function block requests the axis to reverse the direction while the axis is moving. • The required motion profile cannot be achieved due to current velocity too low or too high. Reset the state of the axis using the MC_Reset function block. Correct the motion profile in the function block, or re-execute the function block when the axis velocity is compatible with the requested motion profile.
10	MC_FB_ERR_SOFT_LIMIT	This function block cannot execute as it will end up moving beyond the soft limit, or the function block is aborted as the soft limit has been reached. Check the velocity or target position settings in the function block, or adjust soft limit setting.	The axis is not operational due to soft limit error detected, or due to expected soft limit error in a function block. Reset the state of the axis using the MC_Reset function block. Check the velocity or target position settings for the function block, or adjust Soft Limit setting.
11	MC_FB_ERR_HARD_LIMIT	This function block is aborted as the Hard Limit switch active state has been detected during axis movement, or aborted as the Hard Limit switch active state has been detected before axis movement starts. Move the axis away from the hard limit switch in the opposite direction.	The axis is not operational due to hard limit error detected. Reset the state of the axis using the MC_Reset function block, and then move the axis away from the hard limit switch in the opposite direction.
12	MC_FB_ERR_LOG_LIMIT	This function block cannot execute as it will end up moving beyond the PTO Accumulator logic limit, or the function block is aborted as the PTO Accumulator logic limit has been reached. Check the velocity or target position settings for the function block. Or, use MC_SetPosition function block to adjust the axis coordinate system.	The axis is not operational due to PTO Accumulator logic limit error detected, or due to expected PTO accumulator logic limit error in a function block. Reset the state of the axis using the MC_Reset function block. Check the velocity or target position settings for the function block. Or, use MC_SetPosition function block to adjust the axis coordinate system.
13	MC_FB_ERR_ENGINE	A motion engine execution error is detected during the execution of this function block. Power cycle the whole motion setup, including controller, drives and actuators, and re-download the User Application. If the fault is persistent, call Tech support.	The axis is not operational due to a motion engine execution error. Power cycle the whole motion setup, including controller, drives and actuators, and re-download the User Application. If the fault is persistent, contact your local Rockwell Automation technical support representative. For contact information, see: http://support.rockwellautomation.com/MySupport.asp.
16	MC_FB_ERR_NOT_HOMED	The Function Block cannot execute because the axis needs to be homed first. Execute homing against the axis using MC_Home Function Block.	The axis is not operational because the axis is not homed. Reset the state of the axis using the MC_Reset Function Block.
128	MC_FB_PARAM_MODIFIED	**Warning**: The requested motion parameter for the axis has been adjusted. The function block executes successfully.	Motion internal Fault, Error ID = 0x80. Contact your local Rockwell Automation technical support representative. For contact information, see: http://support.rockwellautomation.com/MySupport.asp.

[1] You can view axis status through the Axis Monitor feature of the Connected Components Workbench software.

When a motion control function block ends with an error, and the axis is in ErrorStop state, in most cases, MC_Reset function block (or, MC_Power Off/On and MC_Reset) can be used to have the axis to be recovered. With this, the axis can get back to normal motion operation without stopping the controller operation.

Major Fault Handling

In case the controller encounters issues where recovery is not possible through the Stop, Reset, or Power function blocks, controller operation will be stopped and a major fault will be reported.

The following motion-related major fault codes are defined for Micro830 and Micro850 controllers.

Major Fault Error Codes and Description

Major Fault Value	Fault ID MACRO	Major Fault description
0xF100	EP_MC_CONFIG_GEN_ERR	There is general configuration error detected in the motion configuration downloaded from Connected Components Workbench, such as Num of Axis, or Motion execution interval being configured out of range. When this major fault is reported, there could be no axis in ErrorStop state.
0xF110	EP_MC_RESOURCE_MISSING	Motion configuration has mismatch issues with motion resource downloaded to the controller. There are some motion resources missing. When this major fault is reported, there could be no axis in ErrorStop state.
0xF12x	EP_MC_CONFIG_AXS_ERR	Motion configuration for axis cannot be supported by this catalog, or the configuration has some resource conflict with some other motion axis, which has been configured earlier. The possible reason could be maximum velocity, max acceleration is configured out of supported range. x = the logic Axis ID (0…3).
0xF15x	EP_MC_ENGINE_ERR	There is a motion engine logic error (firmware logic issue or memory crash) for one axis detected during motion engine cyclic operation. One possible reason can be motion engine data/memory crash. (This is motion engine operation error, and should not happen in normal condition.) x = the logic Axis ID (0…3).

Motion Axis Configuration in Connected Components Workbench

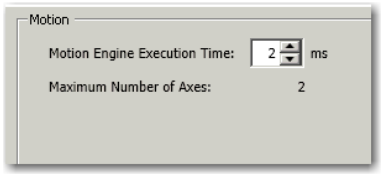
A maximum of three motion axes can be configured through the Connected Components Workbench software. To add, configure, update, delete, and monitor an axis in Connected Components Workbench, refer to the next sections.

TIP Configuration changes must be compiled and downloaded to the controller to take effect.

TIP Values for the different motion axis parameters are validated based on a set of relationships and pre-determined absolute range. See Motion Axis Parameter Validation on page 115 for a description of the relationships between parameters.

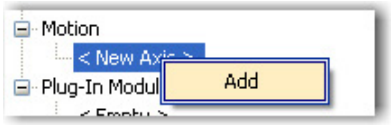
Add New Axis

IMPORTANT Motion Engine Execution Time



When an axis is added to the configuration, the Motion Engine Execution Time can be configured from 1…10 ms (default: 1 ms). This global parameter applies to all motion axis configurations.

1. On the Device Configuration tree, right-click <New Axis>. Click Add.



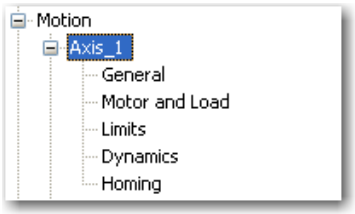
2. Provide an axis name. Click Enter.

TIP Name must begin with a letter or underscore character, followed by a letter or single underscore characters.

TIP You can also press F2 to edit axis name.

3. Expand the newly created Axis to see the following configuration categories:

- General
- Motor and Load
- Limits
- Dynamics
- Homing

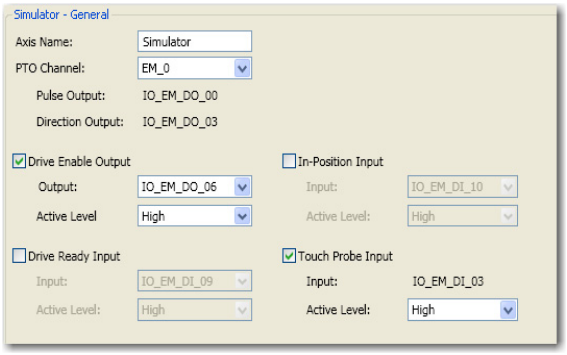


TIP To help you edit these motion properties, see Edit Axis Configuration on page 105. You can also learn more about axis configuration parameters.

Edit Axis Configuration

General Parameters

1. On the axis configuration tree, click General.
   The <Axis Name> - General properties tab appears.



2. Edit General parameters. You can refer to the table for a description of the general configuration parameters for a motion axis.

IMPORTANT To edit these general parameters, you can refer to Input and Output Signals on page 78 for more information about fixed and configurable outputs.

General Parameters

Parameter	Description and Values
Axis Name	User defined. Provides a name for the motion axis.
PTO Channel	Shows the list of available PTO channels.
Pulse output	Presents the logical variable name of the Direction Output channel based on the PTO channel value that has been assigned.
Direction output	Presents the logical variable name of the Direction Output channel based on the PTO channel value that has been assigned.
Drive Enable Output	Servo On Output Enable flag. Check the option box to enable.
- Output	The list of available digital output variables that can be assigned as servo/drive output.
- Active Level	Set as High (default) or Low.
In-position Input	Check the option box to enable in-position input monitoring.
- Input	List of digital input variables for in-position input monitoring. Select an input.

General Parameters

Parameter	Description and Values
- Active Level	Set as High (default) or Low.
Drive ready input	Servo Ready Input Enable flag. Check the option box to enable the input.
- Input	The list of digital input variables. Select an input.
- Active Level	Set as High (default) or Low.
Touch probe input	Configure whether an input for touch probe is used. Check the option box to enable touch probe input.
- Input	List of digital input variables. Select an input
- Active Level	Set the active level for touch probe input as High (default) or Low.

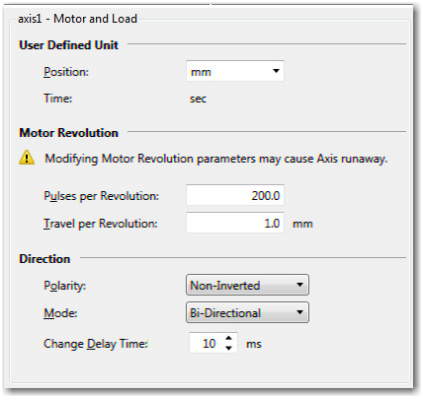
PTO Channel Naming

Names of embedded PTO channels have the prefix EM (embedded) and each available PTO channel is enumerated starting from 0. For example, a controller that supports three axes will have the following PTO channels available:

- EM_0
- EM_1
- EM_2

Motor and Load

Edit the Motor Load properties as defined in the table.



IMPORTANT Certain parameters for Motor and Load are Real values. For more information, see Real Data Resolution on page 112

Motor and Load Parameters

Parameter	Description and Values
User-defined unit	Defines user unit scaling that matches your mechanical system values. These units shall be carried forward into all command and monitor axis in user unit values throughout programming, configuration and monitoring functions.
Position	Select from any of the following options: – mm – cm – inches – revs – custom unit (ASCII format of up to 7 characters long)
Time	Read only. Predefined in seconds.
Motor revolution	Defines pulse per revolution and travel per revolution values.
Pulse per revolution(1)	Defines the number of pulses needed to obtain one revolution of the drive motor. *Range*: 0.0001…8388607 *Default*: 200.0
Travel per revolution(1)	Travel per revolution defines the distance, either linear or rotational, that the load moves per revolution of the motor. *Range*: 0.0001…8388607. *Default*:1.0 user unit.
Direction	Defines polarity, mode, and change of delay time values.
Polarity	Direction polarity determines whether the direction signal received by the controller as a discrete input should be interpreted on the input as received by the motion controller, (that is, the non-inverted case), or whether the signal should be inverted prior to interpretation by the motion control logic. Set as Inverted or Non-inverted (default).
Mode	Set as Bi-directional (default), Positive (clockwise), or Negative (counter-clockwise) direction.
Change delay time	Configure from 0…100 ms. Default value is 10 ms.

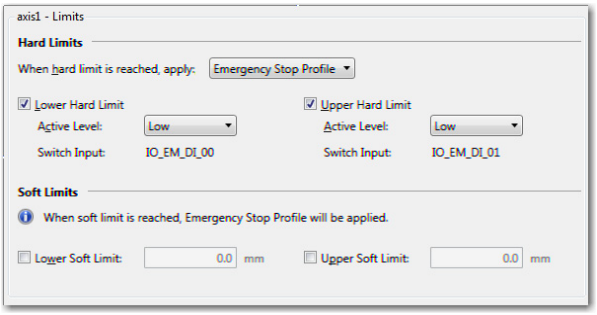
(1) The parameter is set as REAL (float) value in Connected Components Workbench. To learn more about conversions and rounding of REAL values, see Real Data Resolution on page 112.

TIP A red border on an input field indicates that an invalid value has been entered. Scroll over the field to see tooltip message that will let you know the valid value range for the parameter. Supply the valid value.

ATTENTION: Modifying Motor Revolution parameters may cause axis runaway.

Limits

Edit the Limits parameters based on the table below.



ATTENTION: To learn more about the different types of Limits, see Limits on page 94.

Limits Parameters

Parameter[1]	Value
Hard Limits	Defines upper and lower hard limits for the axis.
When hard limits is reached, apply	Configure whether to perform a forced PTO hardware stop (immediately turn off pulse output) or whether to decelerate (leave pulse output on and use deceleration values as defined on the Emergency Stop profile). Set as any of the following: • Forced PTO Hardware Stop • Emergency Stop Profile
Lower Hard Limit	Click checkbox to enable a lower hard limit.
Active Level (for Lower Hard Limit)	High or Low.
Upper Hard Limit	Click checkbox to enable.
Active Level (for Upper Hard Limit)	High or Low.
Soft Limits	Defines upper and lower soft limits values.
Lower Soft Limit[2]	Lower soft limit should be less than upper soft limit. 1. Click checkbox to enable an lower/upper soft limit. 2. Specify a value (in mm).
Upper Soft Limit[2]	

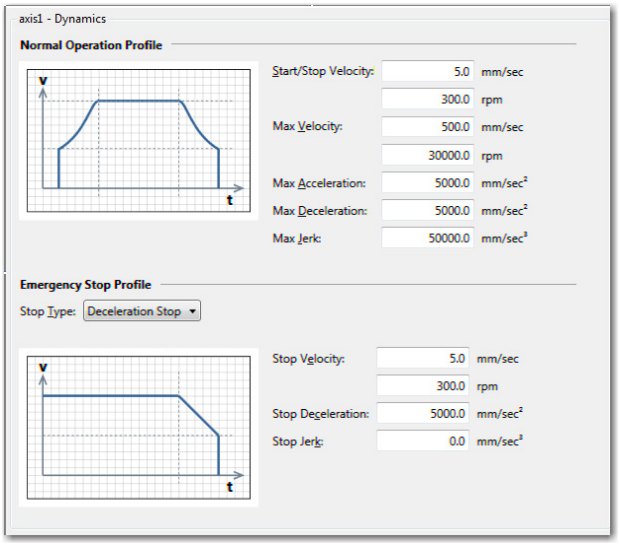
(1) To convert from user units to pulse:

$$\text{Value in user unit} = \text{Value in pulse} \times \frac{\text{Travel per revolution}}{\text{Pulse per revolution}}$$

(2) The parameter is set as REAL (float) value in Connected Components Workbench. To learn more about conversions and rounding of REAL values, see Real Data Resolution on page 112.

TIP A red border on an input field indicates that an invalid value has been entered. Scroll over the field to see tooltip message that will let you know the valid value range for the parameter. Supply the valid value.

3. Click Dynamics. The <Axis Name> - Dynamics tab appears. Edit the Dynamics parameters based on the table below.



Dynamics Parameters

Parameter	Values
Start/Stop Velocity[1] [2]	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range* 1…100,000 pulse/sec *Default*: 300 rpm For example, you can configure the value from 0.005…500 mm/s for 200 pulses per revolution and units of 1 mm per revolution.[3] Rpm value is automatically populated when a value in user units is specified, but the user can also initially enter an rpm value. Start/stop velocity should not be greater than maximum velocity.
Start/Stop Velocity in rpm[1] [2]	
Max Velocity[1] [2]	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 1…10,000,000 pulse/sec. *Default*: 100,000.0 pulse/sec
Max Acceleration[1]	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 1…10,000,000 pulse/sec^2 *Default*: 10,000,000 pulse/sec^2
Max Deceleration[1]	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 1…100,000 pulse/sec^2 *Default*: 10,000,000 pulse/sec^2
Max Jerk[1]	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 0…10,000,000 pulse/sec^3 *Default*: 10,000,000 pulse/sec^3
Emergency Stop Profile	Defines stop type, velocity, deceleration and jerk values.
Stop Type	Set as Deceleration Stop (default) or Immediate Stop.

Dynamics Parameters

Parameter	Values
Stop Velocity(1)	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 1…100,000 pulse/sec *Default*: 300 rpm
Stop Deceleration(1)	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 1…10,000,000 pulse/sec *Default*: 300.0 rpm^2
Stop Jerk(1)	The range is based on Motor and Load parameters (See Motor and Load Parameters on page 107) using: *Range*: 0...10,000,000 pulse/sec^3 *Default*: 0.0 rpm^3 (Disabled)

(1) The parameter is set as REAL (float) value in Connected Components Workbench. To learn more about conversions and rounding of REAL values, see Real Data Resolution on page 112.

(2) The formula for deriving rpm to user unit, and vice versa:

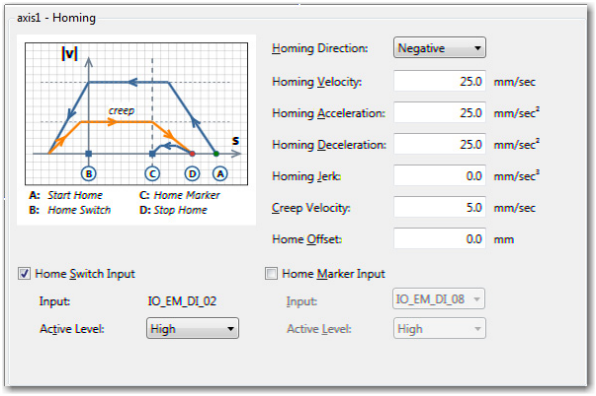
$$v\ (\text{in rpm}) = \frac{v\ (\text{in user unit/sec}) \times 60\ s}{\text{travel per revolution (in user unit)}}$$

(3) To convert from parameter value from pulse to user units:

$$\text{Value in user unit} = \text{Value in pulse} \times \frac{\text{Travel per revolution}}{\text{Pulse per revolution}}$$

TIP A red border on an input field indicates that an invalid value has been entered. Scroll over the field to see tooltip message that will let you know the valid value range for the parameter. Supply the valid value.

4. Set Homing parameters based on the description below. Click Homing.



Homing Parameters

Parameter	Value range
Homing Direction	Positive (clockwise) or negative (counterclockwise).
Homing Velocity(1)	*Range*: 1…100,000 pulse/sec *Default*: 5,000.0 pulse/sec (25.0 mm/sec) **NOTE**: Homing Velocity should not be greater than the maximum velocity.
Homing Acceleration(1)	*Range*: 1…10,000,000 pulse/sec^2 *Default*: 5000.0 pulse/sec^2 (25.0 mm/sec^2) **NOTE**: Homing Acceleration should not be greater than Maximum Acceleration.
Homing Deceleration(1)	*Range*: 1…10,000,000 pulse/sec^2 *Default*: 5000.0 pulse/sec^2 (25.0 mm/sec^2) **NOTE**: Homing Deceleration should not be greater than Maximum Deceleration.
Homing Jerk(1)	*Range*: 0…10,000,000 pulse/sec^3 *Default*: 0.0 pulse/sec^3 (0.0 mm/sec^3) **NOTE**: Homing Jerk should not be greater than Maximum Jerk.
Creep Velocity(1)	*Range*: 1…5,000 pulse/sec *Default*: 1000.0 pulse/sec (5.0 mm/sec) **NOTE**: Homing Creep Velocity should not be greater than Maximum Velocity.
Homing Offset(1)	*Range*: -1073741824…1073741824 pulse *Default*: 0.0 pulse (0.0 mm)
Home Switch Input	Enable home switch input by clicking the checkbox.
- Input	Read only value specifying the input variable for home switch input.
- Active Level	High (default) or Low.
Home Marker Input	Enable the setting of a digital input variable by clicking the checkbox.
- Input	Specify digital input variable for home marker input.
- Active Level	Set the active level for the home switch input as High (default) or Low.

(1) The parameter is set as REAL (float) value in Connected Components Workbench. To learn more about conversions and rounding of REAL values, see Real Data Resolution on page 112.

Axis Start/Stop Velocity

Start/Stop velocity is the initial velocity when an axis starts to move, and the last velocity before the axis stops moving. Generally, Start/Stop velocity is configured at some low value, so that it is smaller than most velocity used in the motion function block.

- When the target velocity is smaller than Start/Stop velocity, move the axis immediately at the target velocity;
- When the target velocity is NOT smaller than Start/Stop velocity, move the axis immediately at Start/Stop velocity;

Real Data Resolution

Certain data elements and axis properties use REAL data format (single-precision floating point format). Real data has seven-digit resolution and digit values entered by the user that are longer than seven digits are converted. See the following examples.

REAL Data Conversion Examples

User value	Converted to
0.12345678	0.1234568
1234.1234567	1234.123
12345678	1.234568E+07 (exponential format)
0.000012345678	1.234568E-05 (exponential format)
2147418166	2.147418+E09
-0.12345678	-0.1234568

If the number of digits is greater than seven (7) and the eighth digit is greater than or equal to 5, then the 7th digit is rounded up. For example:

21474185 rounded to 2.147419E+07
21474186 rounded to 2.147419E+07

If the eighth digit is <5, no rounding is done and the seventh digit remains the same. For example:

21474181 rounded to 2.147418E+07

Examples for Motion Configuration: (1)

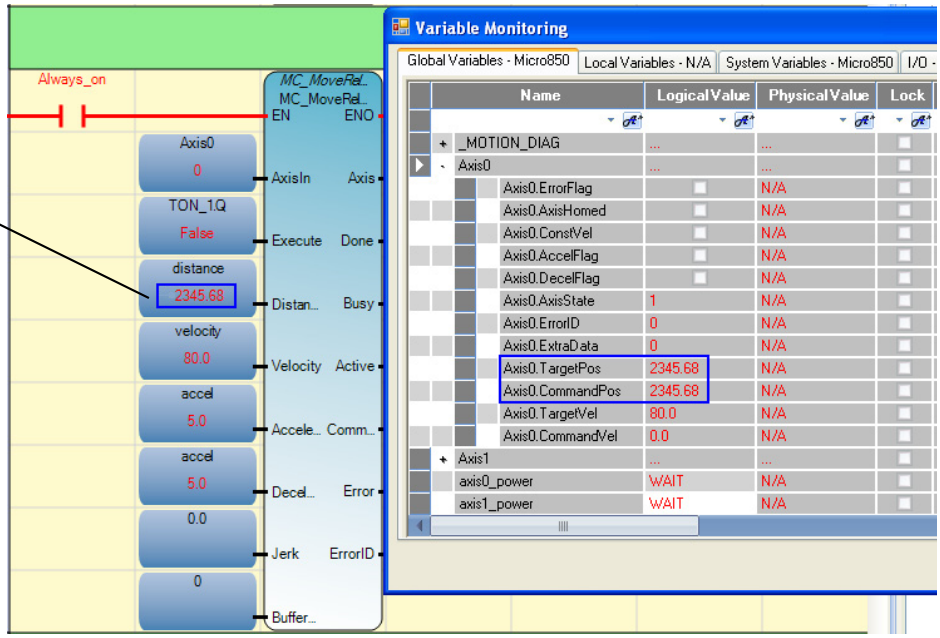
Parameter	Actual Value Entered by User	Converted Value in Connected Components Workbench	Tooltip Error Value(1)
Pulses per revolution	8388608	8388608 (no conversion)	Pulse per revolution must be in the range of 0.0001 to 8388607 user unit.
Upper Soft Limit	10730175	1.073018E+7	Upper Soft limit must be greater than Lower Soft Limit. The range is from 0 (exclusive) to 1.073217E+07 user unit.
Lower Soft Limit	-10730175	-1.073018E+7	Lower Soft limit must be smaller than Upper Soft Limit. The range is from -1.073217E+07 to 0 (exclusive) user unit.

(1) On the axis configuration page in Connected Components Workbench, an input field with a red border indicates that the value that has been entered is invalid. A tooltip message should let you know the expected range of values for the parameter. The range of values presented in the tooltip messages are also presented in REAL data format.

Variable Monitor Example

The Variable Monitor displays six significant digits with rounding, although the real data type still contains seven significant digits.

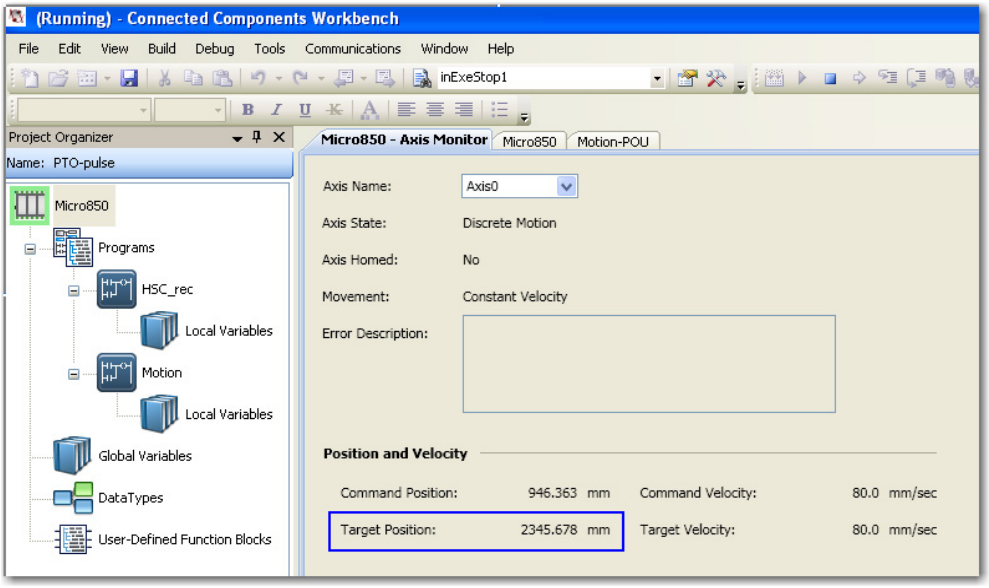
In this example, the user has entered the Target Position value of 2345.678. This value is rounded up to six digits (2345.68) in the Variable Monitoring screen.



(1) For the motion function block parameters, data validation is performed during Run time. The corresponding error will be given if the validation fails.

Axis Monitor Example

The Axis Monitor displays seven significant digits with rounding.

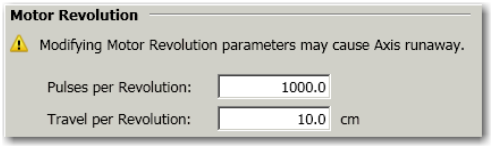


ATTENTION: See Motion Axis Configuration in Connected Components Workbench on page 103 to learn more about the different axis configuration parameters.

PTO Pulse Accuracy

Micro800 motion feature is pulse-based and the value of distance and velocity are designed in such a way that all PTO-related values are integers at the hardware level, when converting to PTO pulse.

For example, if the user configures Motor Pulses per Revolution as 1,000 and Travel per Revolution as 10 cm and the user wants to drive velocity at 4.504 cm/sec. The target velocity is 4.504 cm/sec (that is, 450.4 pulse/sec). In this case, the actual commanded velocity will be 4.5 cm/sec (that is, 450 pulse/sec), and the 0.4 pulse/sec is rounded off.


Motor Revolution
Modifying Motor Revolution parameters may cause Axis runaway.
Pulses per Revolution: 1000.0
Travel per Revolution: 10.0 cm

This rounding scheme also applies to other input parameters such as Position, Distance, Acceleration, Deceleration, and Jerk. For instance, with above motor

revolution configuration, setting Jerk as 4.504 cm/sec^3 is the same as setting Jerk as 4.501 cm/sec^3, as both are rounded off to 4.5 cm/sec^3. This rounding applies to both axis configuration input in the Connected Components Workbench software and function block input.

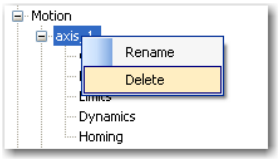
Motion Axis Parameter Validation

Besides falling within the pre-determined absolute range, motion axis parameters are validated based on relationships with other parameters. These relationships or rules are listed below. Error is flagged whenever there is violation to these relationships.

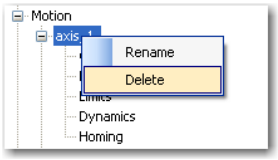
- Lower Soft Limit should be less than the Upper Soft Limit.
- Start/Stop velocity should not be greater than the maximum velocity.
- Emergency Stop velocity should not be greater than the maximum velocity.
- Homing velocity should not be greater than the maximum velocity.
- Homing acceleration should not be greater than maximum acceleration.
- Homing deceleration should not be greater than maximum deceleration.
- Homing jerk should not be greater than maximum jerk.
- Homing creep velocity should not be greater than maximum velocity.

Delete an Axis

1. On the device configuration tree, and under Motion, right-click the axis name and select Delete.



2. A message box appears asking to confirm deletion. Click Yes.



Monitor an Axis

To monitor an axis, the Connected Components Workbench software should be connected to the controller and in DEBUG mode.

1. On the device configuration page, click Axis Monitor.

2. The Axis Monitor window appears with the following characteristics available for viewing:
   - axis state
   - axis homed
   - movement
   - error description
   - command position in user unit
   - command velocity in user unit per second
   - target position in user unit
   - target velocity in user unit per second

Homing Function Block

The homing function block MC_Home commands the axis to perform the "search home" sequence. The Position input is used to set the absolute position when the reference signal is detected, and configured home offset is reached. This function block completes at StandStill if the homing sequence is successful.

MC_Home only can be aborted by the function blocks MC_Stop or MC_Power. Any abort attempt from other moving function blocks will result in function block failure with Error ID = MC_FB_ERR_STATE. However, homing operation is not interrupted, and can be executed as usual.

If MC_Home is aborted before it completes, the previously searched home position is considered as invalid, and the axis Homed status is cleared.

After axis power on is done, the axis Homed status is reset to 0 (not homed). On most scenarios, the MC_Home function block needs to be executed to calibrate the axis position against the axis home configured after MC_Power (On) is done.

There are five homing modes supported on Micro830 and Micro850 controllers.

Homing Modes

Homing Mode Value	Homing Mode name	Homing Mode Description
0x00	MC_HOME_ABS_SWITCH	Homing process searches for Home Absolute switch.
0x01	MC_HOME_LIMIT_SWITCH	Homing process searches for limit switch.
0x02	MC_HOME_REF_WITH_ABS	Homing process searches for Home Absolute switch plus using encoder reference pulse.
0x03	MC_HOME_REF_PULSE	Homing process searches for limit switch plus using encoder reference pulse.
0x04	MC_HOME_DIRECT	Static homing process with direct forcing a home position from user reference. The function block will set current position the mechanism is in as home position, with its position determined by the input parameter, "Position".

IMPORTANT If axis is powered On with only one direction enabled, the MC_Home function block (in modes 0, 1, 2, 3) will generate an error and only MC_Home function block (mode 4) can be executed. See MC_Power function block for more details.

Conditions for Successful Homing

For homing operation to be successful, all configured switches (or sensors) must be properly positioned and wired. The correct position order from the most negative position to the most positive position—that is, from the leftmost to the rightmost in the homing setup diagrams in this section—for the switches are:

1. Lower Limit switch
2. ABS Home switch
3. Upper Limit switch

During MC_Home function block execution, the home position will be reset, and the soft limits mechanical position will be recalculated. During homing sequence, the motion configuration for the soft limits will be ignored.

The homing motion sequence discussed in this section has the following configuration assumptions:

1. Homing direction is configured as negative direction;
2. The Lower limit switch is configured as enabled and wired;

The different homing modes as defined (see table Homing Modes on page 116) can have different, but still similar motion sequence. The concept discussed below is applicable to various homing configurations.

MC_HOME_ABS_SWITCH

IMPORTANT If home switch is not configured as enabled, MC_HOME_ABS_SWITCH (0) homing fails with MC_FB_ERR_PARAM.

MC_HOME_ABS_SWITCH (0) homing procedure performs a homing operation against the home switch. The actual motion sequence is dependent on the home switch, limit switch configuration, and the actual status for the switches before homing starts—that is, when the MC_Home function block is issued.

Scenario 1: Moving part at right (positive) side of home switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to the left side (negative direction);
2. When home switch is detected, the moving part decelerates to stop;
3. Moving part moves back (positive direction) in creep velocity to detect home switch On → Off edge;
4. Once home switch On → Off is detected, record the position as mechanical home position, and decelerate to stop;
5. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis in the Connected Components Workbench software.

Scenario 2: Moving part is in between Lower Limit and Home switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its left side (negative direction);
2. When lower limit switch is detected, the moving part decelerates to stop, or stop immediately, according to limit switch hard stop configuration;
3. Moving part moves back (in positive direction) in creep velocity to detect home switch On → Off edge;
4. Once home switch On → Off edge is detected, record the position as mechanical home position, and decelerate to stop;
5. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis in the Connected Components Workbench software.

TIP If Lower Limit switch is not configured, or not wired, the homing motion fails, and moves continuously to the left until the drive or moving part fails to move.

Scenario 3: Moving part on Lower Limit or Home switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its right side (in positive direction) in creep velocity to detect home switch On → Off edge;
2. Once home switch On → Off edge is detected, record the position as mechanical home position, and decelerate to stop;

3. Move to the configured home position. The mechanical home position recorded during moving right sequence, plus the home offset configured for the axis in the Connected Components Workbench software.

Scenario 4: Moving part at left (negative) side of Lower Limit switch before homing starts

In this case, the homing motion fails and moves continuously to the left until drive or moving part fails to move. User needs to make sure the moving part at the proper location before homing starts.

MC_HOME_LIMIT_SWITCH

IMPORTANT If Lower Limit switch is not configured as Enabled, MC_HOME_LIMIT_SWITCH (1) homing will fail (Error ID: MC_FB_ERR_PARAM).

For Homing against Lower Limit switch, one positive home offset can be configured; for Homing against Upper Limit switch, one negative home offset can be configured.

MC_HOME_LIMIT_SWITCH (1) homing procedure performs a homing operation against Limit switch. The actual motion sequence is dependent on the limit switch configuration and the actual status for the switch before homing starts—that is, when the MC_Home function block is issued.

Scenario 1: Moving part at right (positive) side of Lower Limit switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its left side (in negative direction);
2. When Lower Limit switch is detected, the moving part decelerates to stop, or stops immediately, according to Limit Switch Hard Stop configuration;
3. Moving part moves back (in positive direction) in creep velocity to detect Lower Limit switch On → Off edge;
4. Once Lower Limit switch On → Off edge is detected, record the position as mechanical home position, and decelerate to stop;
5. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis through the Connected Components Workbench software.

Scenario 2: Moving part on Lower Limit switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its right side (in positive direction) in creep velocity to detect Lower Limit switch On → Off edge;
2. Once Lower Limit switch On → Off edge is detected, record the position as mechanical home position, and decelerate to stop;
3. Move to the configured home position. The mechanical home position recorded during moving right sequence, plus the home offset configured for the axis through the software.

Scenario 3: Moving part at left (negative) side of Lower Limit switch before homing starts

In this case, the homing motion fails and moves continuously to the left until drive or moving part fails to move. User needs to make sure the moving part is at the proper location before homing starts.

MC_HOME_REF_WITH_ABS

IMPORTANT If Home switch or Ref Pulse is not configured as Enabled, MC_HOME_REF_WITH_ABS (2) homing fails with Error ID: MC_FB_ERR_PARAM.

MC_HOME_REF_WITH_ABS (2) homing procedure performs a homing operation against Home switch, plus fine Ref Pulse signal. The actual motion sequence is dependent on the home switch, limit switch configuration, and the actual status for the switches before homing starts—that is, when the MC_Home function block is issued.

Scenario 1: Moving part at right (positive) side of Home switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its left side (in negative direction);
2. When Home Abs switch is detected, the moving part decelerates to stop;
3. Moving part moves back (in positive direction) in creep velocity to detect Home Abs On → Off edge;
4. Once Home Abs switch On → Off is detected, start to detect first Ref Pulse signal coming in;
5. Once the first Ref Pulse signal comes, record the position as mechanical home position, and decelerate to stop;
6. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis through the Connected Components Workbench software.

Scenario 2: Moving part between Lower Limit and Home switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its left side (in negative direction);
2. When Lower Limit switch is detected, the moving part decelerates to stop, or stops immediately, according to Limit Switch Hard Stop configuration;
3. Moving part moves back (in positive direction) in creep velocity to detect Home switch On → Off edge;
4. Once Home Abs switch On → Off is detected, start to detect first Ref Pulse signal;
5. Once the first Ref Pulse signal comes, record the position as mechanical home position, and decelerate to stop.
6. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis through the Connected Components Workbench software.

IMPORTANT In this case, if Lower limit switch is not configured, or not wired, the homing motion will fail and moves continuously to the left until the drive or moving part fails to move.

Scenario 3: Moving part on Lower Limit or Home switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its right side (in positive direction) in creep velocity to detect Home switch On → Off edge;
2. Once Home Abs switch On → Off is detected, start to detect first Ref Pulse signal;
3. Once the first Ref Pulse signal comes, record the position as mechanical home position, and decelerate to stop;
4. Move to the configured home position. The mechanical home position recorded during moving right sequence, plus the home offset configured for the axis in the Connected Components Workbench software.

Scenario 4: Moving part at left (negative) side of Lower Limit switch before homing starts

In this case, the homing motion fails and moves continuously to the left until drive or moving part fails to move. User needs to make sure the moving part is at the proper location before homing starts.

MC_HOME_REF_PULSE

IMPORTANT If Lower Limit switch or Ref Pulse is not configured as Enabled, MC_HOME_REF_PULSE (3) homing fails (ErrorID: MC_FB_ERR_PARAM).

For Homing against Lower Limit switch, one positive home offset can be configured; for Homing against Upper Limit switch, one negative home offset can be configured.

MC_HOME_REF_PULSE (3) homing procedure performs a homing operation against Limit switch, plus fine Ref Pulse signal. The actual motion sequence is dependent on the limit switch configuration, and the actual status for the switches before homing starts—that is, when the MC_Home function block is issued.

Scenario 1: Moving part at right (positive) side of Lower Limit switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its left side (in negative direction);
2. When Lower Limit switch is detected, the moving part decelerates to stop, or stops immediately, according to Limit Switch Hard Stop configuration;
3. Moving part moves back (in positive direction) in creep velocity to detect Lower Limit switch On → Off edge;
4. Once Lower Limit switch On → Off edge is detected, start to detect first Ref Pulse signal;
5. Once the first Ref Pulse signal comes, record the position as the mechanical home position, and decelerate to stop;
6. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis through the Connected Components Workbench software.

Scenario 2: Moving part on Lower Limit switch before homing starts

The homing motion sequence for this scenario is as follows:

1. Moving part moves to its right side (in Positive direction) in creep velocity to detect Lower Limit switch On → Off edge;
2. Once Lower Limit switch On → Off edge is detected, start to detect first Ref Pulse signal;
3. Once the first Ref Pulse signal comes, record the position as the mechanical home position, and decelerate to stop;

4. Move to the configured home position. The mechanical home position recorded during moving back sequence, plus the home offset configured for the axis through the Connected Components Workbench software.

Scenario 3: Moving part at left (negative) side of Lower Limit switch before homing starts

In this case, the homing motion fails and moves continuously to the left until drive or moving part fails to move. User needs to make sure the moving part at the proper location before homing starts.

MC_HOME_DIRECT

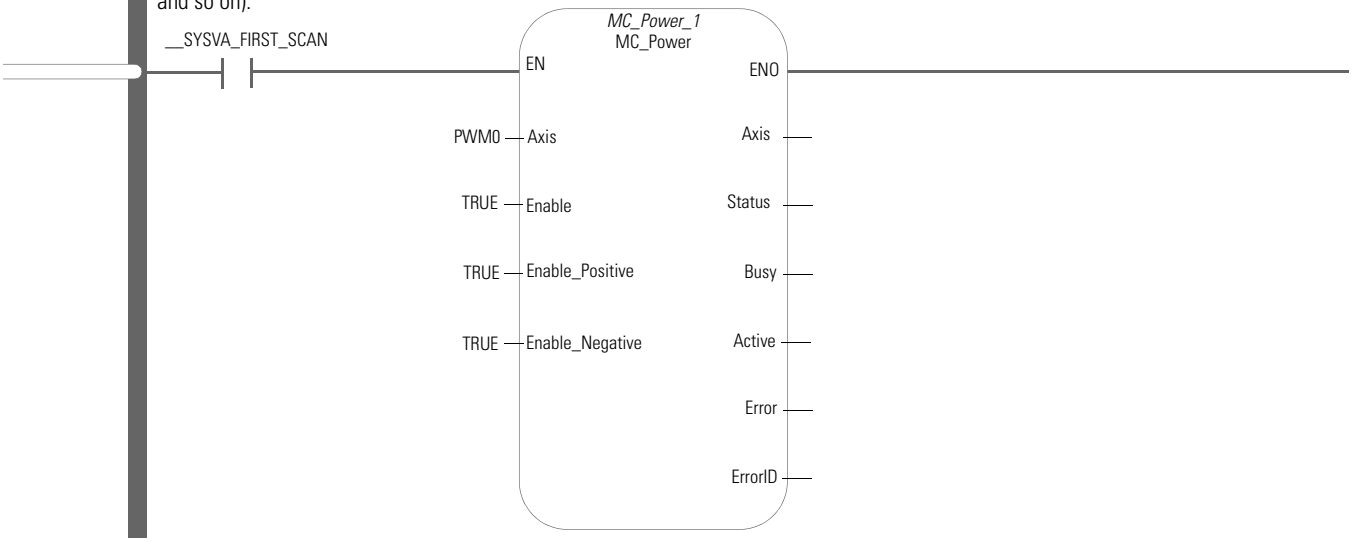
MC_HOME_DIRECT (4) homing procedure performs a static homing by directly forcing an actual position. No physical motion is performed in this mode. This is equivalent to a MC_SetPosition action, except that Axis Homed status will be on once MC_Home (mode = 4) is performed successfully.

Use PTO for PWM Control

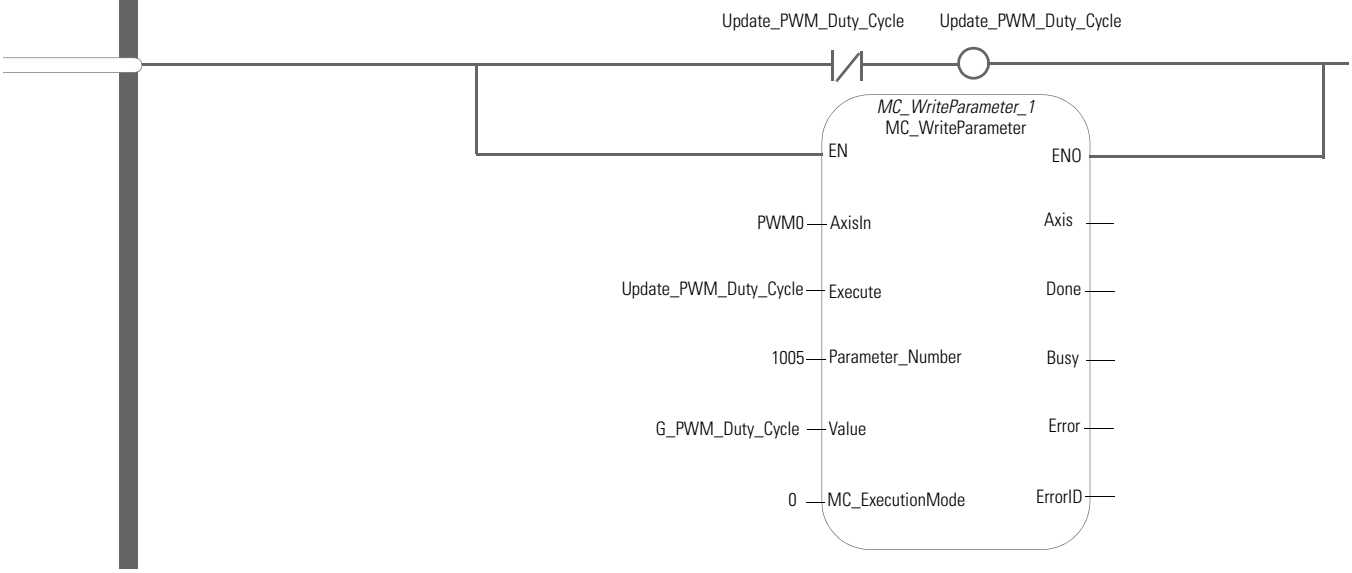
The following example shows you how to use a PTO axis as a PWM.

Launch Connected Components Workbench and create the following ladder program.

1 Enable/power up the PWM axis immediately after going to RUN mode. PWM axis will remain powered ON (until Program mode, and so on).

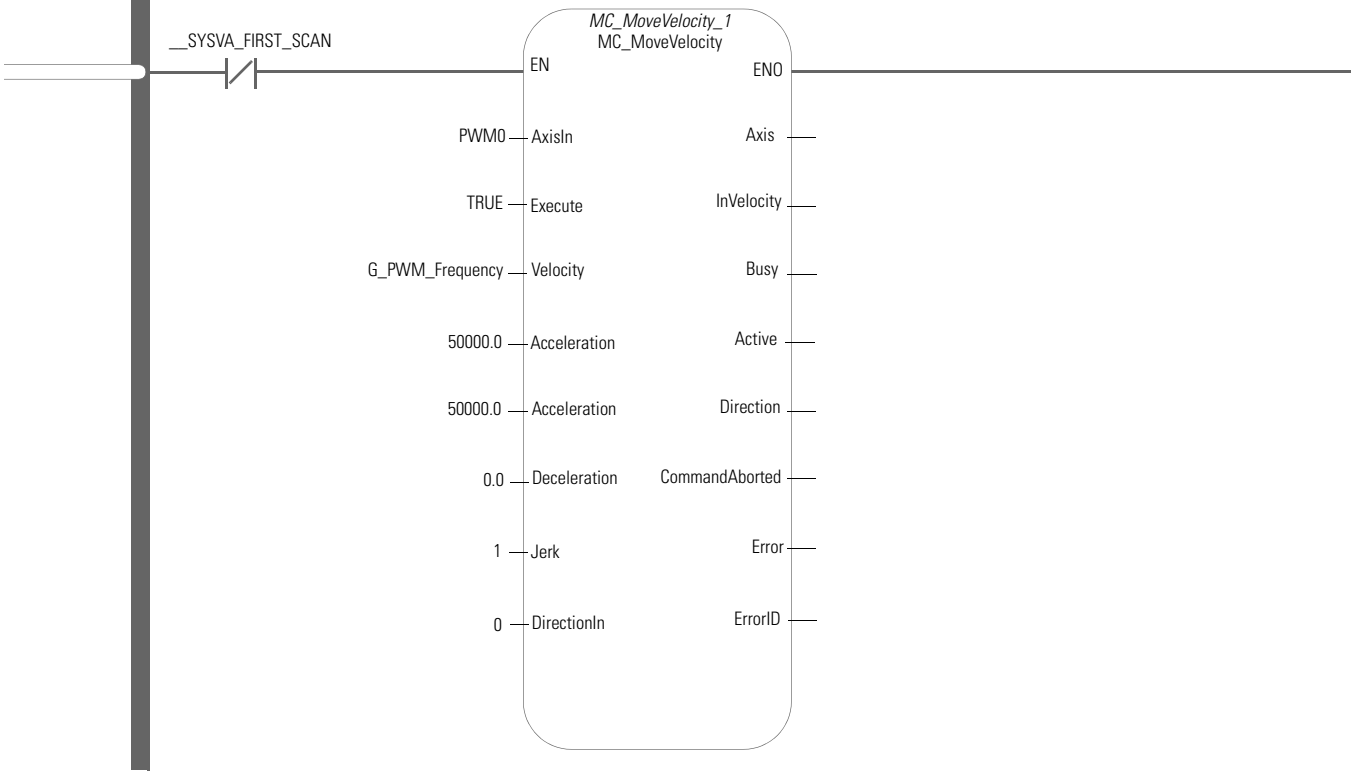


2 Continually use MC_WriteParameter (Parameter 1005) to change the Duty Cycle from global variable G_PWM_Duty_Cycle (example: 0.5 => 50%)



3

After first scan, use MC_MoveVelocity to continually set the PWM frequency (for example: 50,000 => 50 KHz) from global variable G_PWM_Frequency. PWM axis will run forever (until Program Mode, MC_Halt, and so on).



POU PWM_Program

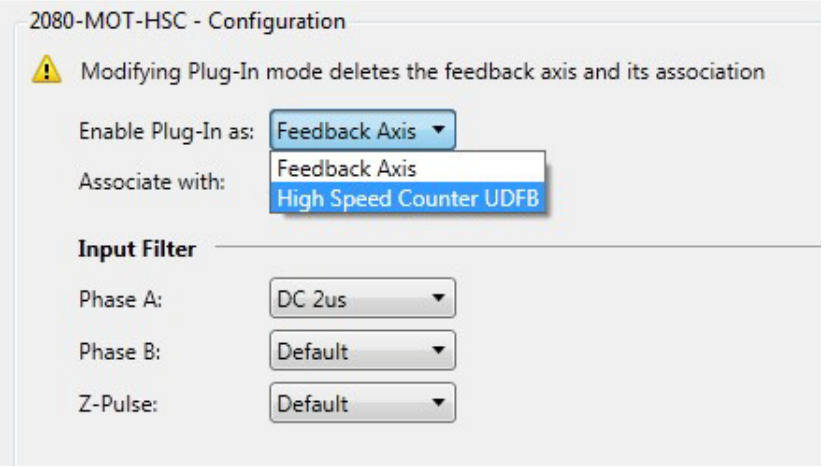
The POU defines four variables.

Variable MC_Power_1 (* *) Direction: VAR Data Type: MC_Power Attribute: ReadWrite Direct variable (Channel):	**Variable MC_MoveVelocity_1** (* *) Direction: VAR Data Type: MC_MoveVelocity Attribute: ReadWrite Direct variable (Channel):
Variable Update_PWM_Duty_Cycle (* *) Direction: Var Data type: BOOL Attribute: ReadWrite Direct variable (Channel):	**Variable MC_Power_1** (* *) Direction: VAR Data Type: MC_Power Attribute: ReadWrite Direct variable (Channel):

HSC Feedback Axis

From Connected Components Workbench Release 8.0 onwards, support has been added for a HSC (High Speed Counter) Feedback Axis which uses the same instructions as the PTO Motion Axis. UDFBs are still supported (you can use either one but you cannot select both for the same plug-in).

Example of Selecting Feedback Axis or UDFB with 2080-MOT-HSC Plug-in



The HSC Feedback Axis provides ease-of-use as you no longer need to program the function blocks, and it also uses up less memory on the controller. The HSC Feedback Axis uses only the administrative function blocks from the PTO Motion Axis and they share the same Axis Monitor.

IMPORTANT	The counters are not reset to zero for program download. For example, if using the feedback axis, use the MC_ResetPosition function block to reset the position to zero.

IMPORTANT	If the feedback axis is in the error state because the configured position limits have been exceeded, using the MC_Reset function block to reset the axis may not clear the error as there may still be pulse detected from the encoder.

Use the High-Speed Counter and Programmable Limit Switch

High-Speed Counter Overview

All Micro830 and Micro850 controllers, except for 2080-LCxx-AWB, support up to six high speed counters (HSC). The HSC feature in Micro800 consists of two main components: the high-speed counter hardware (embedded inputs in the controller), and high-speed counter instructions in the application program. High-speed counter instructions apply configuration to the high-speed counter hardware and updates the accumulator.

ATTENTION: To use the Micro800 HSC feature effectively, you need to have a basic understanding of the following:

- HSC components and data elements.
  The first sections of the chapter provides a detailed description of these components. Quickstart instructions (see page 201) are also available to guide you through setting up a sample HSC project.
- Programming and working with elements in Connected Components Workbench.
  The user needs to have a working knowledge of programming through ladder diagram, structured text, or function block diagram to be able to work with the HSC function block and variables.

ATTENTION: Additional information is available on the HSC function block and its elements in the Connected Components Workbench Online Help that comes with your Connected Components Workbench installation.

This chapter describes how to use the HSC function and also contains sections on the HSC and HSC_ SET_STS function blocks, as follows:

- High Speed Counter (HSC) Data Structures
- HSC (High Speed Counter) Function Block
- HSC_SET_STS Function Block
- Programmable Limit Switch (PLS) Function
- HSC Interrupts

Programmable Limit Switch Overview

The Programmable Limit Switch function allows you to configure the High-Speed Counter to operate as a PLS (Programmable Limit Switch) or rotary cam switch. For more information, see Programmable Limit Switch (PLS) Function on page 153.

What is High-Speed Counter?

High-Speed Counter is used to detect narrow (fast) pulses, and its specialized instructions to initiate other control operations based on counts reaching preset values. These control operations include the automatic and immediate execution of the high-speed counter interrupt routine and the immediate update of outputs based on a source and mask pattern you set.

The HSC functions are different than most other controller instructions. Their operation is performed by custom circuitry that runs in parallel with the main system processor. This is necessary because of the high performance requirements of these functions.

Features and Operation

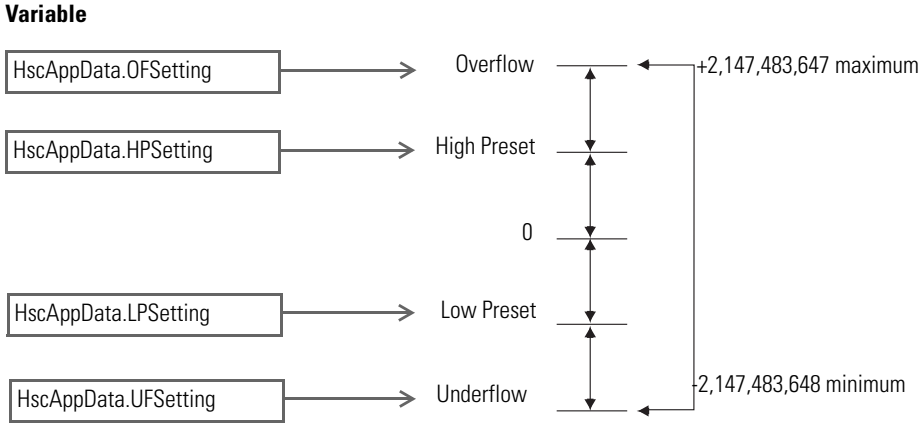
The HSC is extremely versatile; you can select or configure the master HSC for any one of ten (10) modes and the sub HSC for any one of five (5) modes of operation. See HSC Mode (HSCAPP.HSCMode) on page 134 for more information.

Some of the enhanced capabilities of the High-Speed Counters are:

- 100 kHz operation
- Direct control of outputs
- 32-bit signed integer data (count range of ± 2,147,483,647)
- Programmable High and Low presets, and Overflow and Underflow setpoints
- Automatic Interrupt processing based on accumulated count
- Change parameters on-the-fly (from the user control program)

The High-Speed Counter function operates as described in the following diagram.

High Speed Counter Operation



TIP You must set a proper value for the variables OFSetting, HPSetting, and UFSetting before triggering Start/Run HSC. Otherwise, the controller will be faulted. (Setting a value for LPSetting is optional for certain counting modes.)

To learn more about HscAppData variable input, see HSC APP Data Structure on page 133.

When using HSC function blocks, it is recommended that you:

- set HSCAppData underflow setting (UFSetting) and low preset setting (LPSetting) to a value less than 0 to avoid possible HSC malfunction when the HSC accumulator is reset to 0.
- set HSCAppData overflow setting (OFSetting) and high preset setting (HPSetting) to a value greater than 0 to avoid possible HSC malfunction when the HSC accumulator is reset to 0.

In some cases, a sub counter will be disabled by master counter mode. See the section HSC Mode (HSCAPP.HSCMode) on page 134.

TIP HSC0 is used in this document to define how any HSC works.

IMPORTANT The HSC function can only be used with the controller's embedded I/O. It cannot be used with expansion I/O modules.

HSC Inputs and Wiring Mapping

All Micro830 and Micro850 controllers, except 2080-LCxx-xxAWB, have 100 kHz high-speed counters. Each main high-speed counter has four dedicated inputs and each sub high-speed counter has two dedicated inputs.

Micro830 and Micro850 High Speed Counters

	10/16-point	24-point	48-point
Number of HSC	2	4	6
Main high-speed counters	1 (counter 0)	2 (counter 0,2)	3 (counters 0, 2 and 4)
Sub high-speed counters	1 (counter 1)	2 (counter 1,3)	3 (counters 1, 3 and 5)

High Speed Counter	Inputs used
HSC0	0, 1, 2, 3
HSC1	2, 3
HSC2	4, 5, 6, 7
HSC3	6, 7
HSC4	8, 9, 10, 11
HSC5	10, 11

HSC0's sub counter is HSC1, HSC2's sub counter is HSC3 and HSC4's sub counter is HSC5. Each set of counters share the input. The following table shows the dedicated inputs for the HSCs depending on the mode.

HSC Input Wiring Mapping

	Embedded Input											
	0	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**
HSC0	A/C	B/D	Reset	Hold								
HSC1			A/C	B/D								
HSC2					A/C	B/D	Reset	Hold				
HSC3							A/C	B/D				
HSC4									A/C	B/D	Reset	Hold
HSC5											A/C	B/D

The following tables show the input wiring mapping for the different Micro830 and Micro850 controllers.

Micro830 10 and 16-point Controller HSC Input Wiring Mapping

Modes of Operation	Input 0 (HSC0) Input 2 (HSC1)	Input 1 (HSC0) Input 3 (HSC1)	Input 2 (HSC0)	Input 3 (HSC0)	Mode Value in User Program (HSCAppData.HSCMode)
Counter with Internal Direction (mode 1a)	Count Up	Not Used			0
Counter with Internal Direction, External Reset and Hold (mode 1b)	Count Up	Not Used	Reset	Hold	1
Counter with External Direction (mode 2a)	Count Up/Down	Direction	Not Used		2
Counter with External Direction, Reset and Hold (mode 2b)	Count	Direction	Reset	Hold	3
Two Input Counter (mode 3a)	Count Up	Count Down	Not Used		4
Two Input Counter with External Reset and Hold (mode 3b)	Count Up	Count Down	Reset	Hold	5
Quadrature Counter (mode 4a)	A Type input	B Type input	Not Used		6
Quadrature Counter with External Reset and Hold (mode 4b)	A Type input	B Type input	Z Type Reset	Hold	7
Quadrature X4 Counter (mode 5a)	A Type input	B Type input	Not Used		8
Quadrature X4 Counter with External Reset and Hold	A Type input	B Type input	Z Type Reset	Hold	9

Micro830/Micro850 24-point Controller HSC Input Wiring Mapping

Modes of Operation	Input 0 (HSC0) Input 2 (HSC1) Input 4 (HSC2) Input 6 (HSC3)	Input 1 (HSC0) Input 3 (HSC1) Input 5 (HSC2) Input 7 (HSC3)	Input 2 (HSC0) Input 6 (HSC2)	Input 3 (HSC0) Input 7 (HSC2)	Mode Value in User Program
Counter with Internal Direction (mode 1a)	Count Up	Not Used			0
Counter with Internal Direction, External Reset and Hold (mode 1b)	Count Up	Not Used	Reset	Hold	1
Counter with External Direction (mode 2a)	Count Up/Down	Direction	Not Used		2
Counter with External Direction, Reset and Hold (mode 2b)	Count Up/Down	Direction	Reset	Hold	3
Two Input Counter (mode 3a)	Count Up	Count Down	Not Used		4
Two Input Counter with External Reset and Hold (mode 3b)	Count Up	Count Down	Reset	Hold	5
Quadrature Counter (mode 4a)	A Type input	B Type input	Not Used		6
Quadrature Counter with External Reset and Hold (mode 4b)	A Type input	B Type input	Z Type Reset	Hold	7
Quadrature X4 Counter (mode 5a)	A Type input	B Type input	Not Used		8
Quadrature X4 Counter with External Reset and Hold	A Type input	B Type input	Z Type Reset	Hold	9

Micro830/Micro850 48-point Controller HSC Input Wiring Mapping

Modes of Operation	Input 0 (HSC0) Input 2 (HSC1) Input 4 (HSC2) Input 6 (HSC3) Input 8 (HSC4) Input 10 (HSC5)	Input 1 (HSC0) Input 3 (HSC1) Input 5 (HSC2) Input 7 (HSC3) Input 9 (HSC4) Input 11 (HSC5)	Input 2 (HSC0) Input 6 (HSC2) Input 10 (HSC4)	Input 3 (HSC0) Input 7 (HSC2) Input 11 (HSC4)	Mode Value in User Program
Counter with Internal Direction (mode 1a)	Count Up	Not Used			0
Counter with Internal Direction, External Reset and Hold (mode 1b)	Count Up	Not Used	Reset	Hold	1
Counter with External Direction (mode 2a)	Count Up/Down	Direction	Not Used		2
Counter with External Direction, Reset and Hold (mode 2b)	Count Up/Down	Direction	Reset	Hold	3
Two Input Counter (mode 3a)	Count Up	Count Down	Not Used		4
Two Input Counter with External Reset and Hold (mode 3b)	Count Up	Count Down	Reset	Hold	5
Quadrature Counter (mode 4a)	A Type input	B Type input	Not Used		6

Micro830/Micro850 48-point Controller HSC Input Wiring Mapping

Modes of Operation	Input 0 (HSC0) Input 2 (HSC1) Input 4 (HSC2) Input 6 (HSC3) Input 8 (HSC4) Input 10 (HSC5)	Input 1 (HSC0) Input 3 (HSC1) Input 5 (HSC2) Input 7 (HSC3) Input 9 (HSC4) Input 11 (HSC5)	Input 2 (HSC0) Input 6 (HSC2) Input 10 (HSC4)	Input 3 (HSC0) Input 7 (HSC2) Input 11 (HSC4)	Mode Value in User Program
Quadrature Counter with External Reset and Hold (mode 4b)	A Type input	B Type input	Z Type Reset	Hold	7
Quadrature X4 Counter (mode 5a)	A Type input	B Type input	Not Used		8
Quadrature X4 Counter with External Reset and Hold	A Type input	B Type input	Z Type Reset	Hold	9

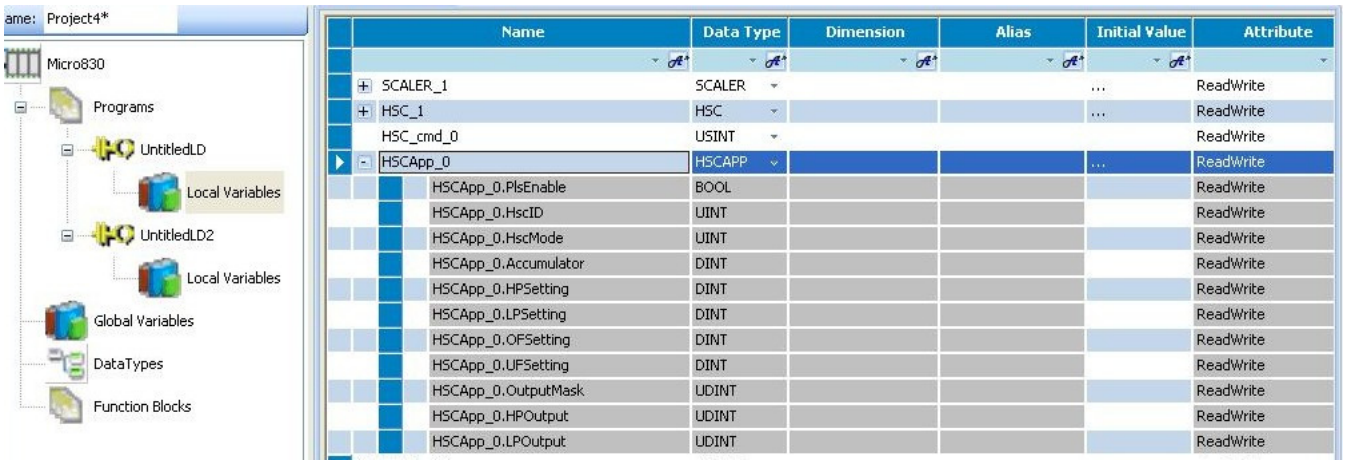
High Speed Counter (HSC) Data Structures

The following section describes HSC data structures.

HSC APP Data Structure

Define a HSC App Data (configuration data, data type HSCAPP) when programming a HSC. During HSC counting, the data should not be changed, except if the configuration needs to be reloaded.

To reload HSC configuration, change the HSC APP Data, then call HSC function block with command 0x03 (set/reload). Otherwise, the change to HSC App Data during HSC counting will be ignored.



TIP HSC1, HSC3, and HSC5 support mode 0, 2, 4, 6, and 8 only, and HSC0, HSC2 and HSC4 support all counting modes.

PLS Enable (HSCAPP.PLSEnable)

Description	Data Format	User Program Access
PLSEnable	bit	read/write

This bit enables and disables the HSC Programmable Limit Switch (PLS) function.

When the PLS function is enabled, the setting in

- HSCAPP.HpSetting
- HSCAPP.LpSetting
- HSCAPP.HPOutput
- HSCAPP.LPOutput

are superseded by corresponding data values from PLS data. See Programmable Limit Switch (PLS) Function on page 153 for more information.

HSCID (HSCAPP.HSCID)

Description	Data Format	User Program Access
HSCID	Word (UINT)	read/write

The following table lists the definition for HSCID.

HSCID Definition

Bits	Description
15…13	HSC Module Type: 0x00: Embedded 0x01: Expansion (not yet implemented) 0x02: Plug-in module
12…8	Module Slot ID: 0x00: Embedded 0x01…0x1F: Expansion (not yet implemented) 0x01…0x05: Plug-in module
7…0	Module internal HSC ID: 0x00-0x0F: Embedded 0x00-0x07: Expansion (not yet implemented) 0x00-0x07: Plug-in module

For Embedded HSC, valid HSCID value is only 0...5.

HSC Mode (HSCAPP.HSCMode)

Description	Data Format	User Program Access
HSC Mode	word (UINT)	read/write

The HSCMode variable sets the High-Speed Counter to one of 10 types of operation. This integer value is configured through the programming device and is accessible in the control program.

HSC Operating Modes

Mode Number	Type
0	Up Counter – The accumulator is immediately cleared (0) when it reaches the high preset. A low preset cannot be defined in this mode.
1	Up Counter with external reset and hold – The accumulator is immediately cleared (0) when it reaches the high preset. A low preset cannot be defined in this mode.
2	Counter with external direction
3	Counter with external direction, reset, and hold
4	Two input counter (up and down)
5	Two input counter (up and down) with external reset and hold
6	Quadrature counter (phased inputs A and B)
7	Quadrature counter (phased inputs A and B) with external reset and hold
8	Quadrature X4 counter (phased inputs A and B)
9	Quadrature X4 counter (phased inputs A and B) with external reset and hold

The main high-speed counters support 10 types of operation mode and the sub high-speed counters support 5 types (mode 0, 2, 4, 6, 8). If the main high-speed counter is set to mode 1, 3, 5, 7 or 9, then the resub high-speed counter will be disabled.

For more information on HSC Function Operating Modes and Input Assignments, see HSC Inputs and Wiring Mapping on page 129.

HSC Mode 0 – Up Counter

HSC Mode 0 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count**				**Not Used**				**Not Used**				**Not Used**					
Example 1	⇑																on (1)	HSC Accumulator + 1 count
Example 2	⇑	on (1)	⇓	off (0)													off (0)	Hold accumulator value

Blank cells = don't care, ⇑= rising edge, ⇓ = falling edge

TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 1 – Up Counter with External Reset and Hold

HSC Mode 1 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count**				**Not Used**				**Reset**				**Hold**					
Example 1	⇑									on (1)	⇓	off (0)				off (0)	on (1)	HSC Accumulator + 1 count
Example 2										on (1)	⇓	off (0)		on (1)				Hold accumulator value
Example3										on (1)	⇓	off (0)					off (0)	Hold accumulator value
Example 4		on (1)	⇓	off (0)						on (1)	⇓	off (0)						Hold accumulator value
Example 5									⇑									Clear accumulator (=0)

Blank cells = don't care, ⇑= rising edge, ⇓ = falling edge

TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 2 – Counter with External Direction

HSC Mode 2 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count**				**Direction**				**Not Used**				**Not Used**					
Example 1	⇑							off (0)									on (1)	HSC Accumulator + 1 count
Example 2	⇑					on (1)											on (1)	HSC Accumulator - 1 count
Example3																	off (0)	Hold accumulator value

Blank cells = don't care, ⇑= rising edge, ⇓ = falling edge

TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 3 – Counter with External Direction, Reset, and Hold

HSC Mode 3 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count**				**Direction**				**Reset**				**Hold**					
Example 1	⇑							off (0)		on (1)	⇓	off (0)				off (0)	on (1)	HSC Accumulator + 1 count
Example 2	⇑					on (1)				on (1)	⇓	off (0)				off (0)	on (1)	HSC Accumulator - 1 count
Example3										on (1)	⇓	off (0)		on (1)				Hold accumulator value
Example 4										on (1)	⇓	off (0)					off (0)	Hold accumulator value
Example 5		on (1)	⇓	off (0)						on (1)	⇓	off (0)						Hold accumulator value
Example 6									⇑									Clear accumulator (=0)

Blank cells = don't care, ⇑= rising edge, ⇓ = falling edge

TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 4 – Two Input Counter (up and down)

HSC Mode 4 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count Up**				**Count Down**				**Not Used**				**Not Used**					
Example 1	⇑					on (1)	⇓	off (0)									on (1)	HSC Accumulator + 1 count
Example 2		on (1)	⇓	off (0)	⇑												on (1)	HSC Accumulator - 1 count
Example3																	off (0)	Hold accumulator value

Blank cells = don't care, ⇑ = rising edge, ⇓ = falling edge

TIP Inputs 0 through 11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 5 – Two Input Counter (up and down) with External Reset and Hold

HSC Mode 5 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count**				**Direction**				**Reset**				**Hold**					
Example 1	⇑					on (1)	⇓	off (0)		on (1)	⇓	off (0)				off (0)	on (1)	HSC Accumulator + 1 count
Example 2		on (1)	⇓	off (0)	⇑					on (1)	⇓	off (0)				off (0)	on (1)	HSC Accumulator - 1 count
Example3										on (1)	⇓	off (0)		on (1)				Hold accumulator value
Example 4										on (1)	⇓	off (0)					off (0)	Hold accumulator value
Example 5		on (1)	⇓	off (0)						on (1)	⇓	off (0)						Hold accumulator value
Example 6									⇑									Clear accumulator (=0)

Blank cells = don't care, ⇑ = rising edge, ⇓ = falling edge

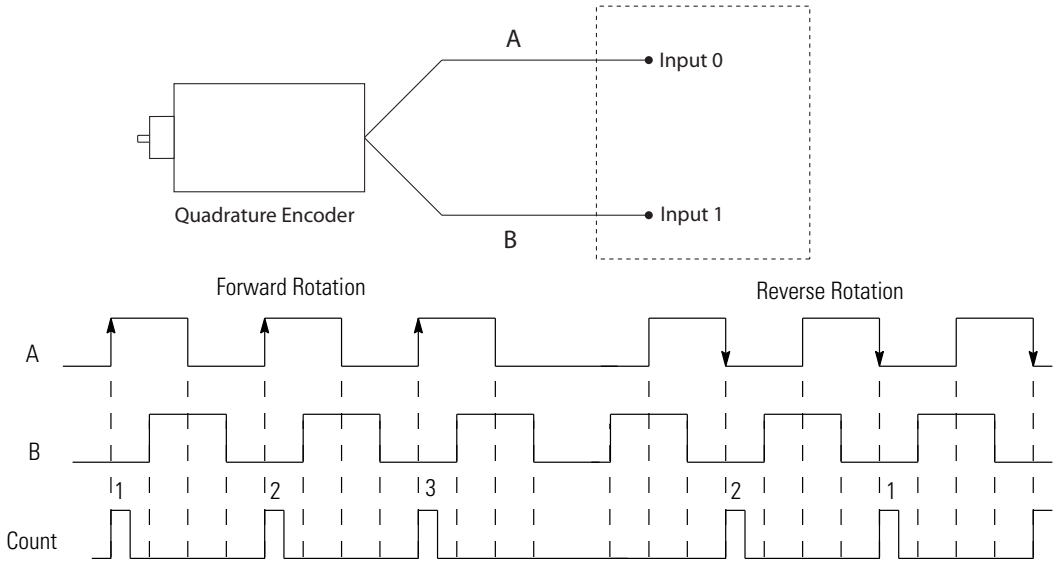
TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

Using the Quadrature Encoder

The Quadrature Encoder is used for determining direction of rotation and position for rotating, such as a lathe. The Bidirectional Counter counts the rotation of the Quadrature Encoder.

The figure below shows a quadrature encoder connected to inputs 0, 1, and 2. The count direction is determined by the phase angle between A and B. If A leads B, the counter increments. If B leads A, the counter decrements.

The counter can be reset using the Z input. The Z outputs from the encoders typically provide one pulse per revolution.



HSC Mode 6 – Quadrature Counter (phased inputs A and B)

HSC Mode 6 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count A**				**Count B**				**Not Used**				**Not Used**					
Example 1[1]	⇑							off (0)									on (1)	HSC Accumulator + 1 count
Example 2[2]			⇓					off (0)									on (1)	HSC Accumulator - 1 count
Example3				off (0)														Hold accumulator value
Example 4		on (1)																Hold accumulator value
Example 5						on (1)												Hold accumulator value
Example 6																	off (0)	Hold accumulator value

(1) Count input A leads count input B.

(2) Count input B leads count input A.
Blank cells = don't care, ⇑ = rising edge, ⇓ = falling edge

TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 7 – Quadrature Counter (phased inputs A and B) With External Reset and Hold

HSC Mode 7 Examples

Input Terminals	Embedded Input 0				Embedded Input 1				Embedded Input 2				Embedded Input 3				**CE Bit**	**Comments**
Function	**Count A**				**Count B**				**Z reset**				**Hold**					
Example 1[1]	⇑							off (0)								off (0)	on (1)	HSC Accumulator + 1 count
Example 2[2]			⇓					off (0)				off (0)				off (0)	on (1)	HSC Accumulator - 1 count
Example3			⇓	off (0)				off (0)		on (1)								Reset accumulator to zero
Example 4		on (1)																Hold accumulator value
Example 5						on (1)												Hold accumulator value
Example 6												off (0)		on (1)				Hold accumulator value
Example 7												off (0)					off (0)	Hold accumulator value

(1) Count input A leads count input B.

(2) Count input B leads count input A.

Blank cells = don't care, ⇑ = rising edge, ⇓ = falling edge

TIP Inputs 0…11 are available for use as inputs to other functions regardless of the HSC being used.

HSC Mode 8 – Quadrature X4 Counter

HSC Mode 8 Examples

Embedded Input 1(HSC0) (A)	**Embedded Input 1(HSC0) (B)**	**Value of CE Bit**	**Accumulator and Counter Action**
▲	OFF	TRUE	Count Up Acc. Value
▲	ON	TRUE	Count Down Acc. Value
▼	OFF	TRUE	Count Down Acc. Value
▼	ON	TRUE	Count Up Acc. Value
OFF	▲	TRUE	Count Down Acc. Value
ON	▲	TRUE	Count Up Acc. Value
OFF	▼	TRUE	Count Up Acc. Value
ON	▼	TRUE	Count Down Acc. Value
OFF or ON	OFF or ON	X	Hold Acc. Value
X	X	FALSE	Hold Acc. Value

HSC Mode 9 – Quadrature X4 Counter with External Reset and Hold

HSC Mode 9 Examples

Embedded Input 0(HSC0) (A))	Embedded Input 1(HSC0) (B)	Embedded Input 2(HSC0) (Reset)	Embedded Input 3(HSC0) (Hold)	Value of CE Bit	Accumulator and Counter Action
▲	OFF	X	-	TRUE	Count Up Acc. Value
▲	ON	X	-	TRUE	Count Down Acc. Value
▼	OFF	X	-	TRUE	Count Down Acc. Value
▼	ON	X	-	TRUE	Count Up Acc. Value
OFF	▲	X	-	TRUE	Count Down Acc. Value
ON	▲	X	-	TRUE	Count Up Acc. Value
OFF	▼	X	-	TRUE	Count Up Acc. Value
ON	▼	X	-	TRUE	Count Down Acc. Value
OFF or ON	OFF or ON	OFF	X	X	Hold Acc. Value
OFF	OFF	ON	X	X	Reset Acc. to Zero
X	X	OFF	ON	X	Hold Acc. Value
X	X	OFF	X	FALSE	Hold Acc. Value

Accumulator (HSCAPP. Accumulator)

Description	Data Format	User Program Access
HSCAPP.Accumulator	long word (32-bit INT)	read/write

This parameter is the initial HSC Accumulator value that need to be set when starting the HSC. This parameter is updated by the HSC sub-system automatically when the HSC is in Counting mode, reflecting the actual HSC accumulator value.

High Preset (HSCAPP.HPSetting)

Description	Data Format	User Program Access
HSCAPP.HPSetting	long word (32-bit INT)	read/write

The HSCAPP.HPSetting is the upper setpoint (in counts) that defines when the HSC sub-system generates an interrupt.

The data loaded into the high preset must be less than or equal to the data resident in the overflow (HSCAPP.OFSetting) parameter or an HSC error is generated.

Low Preset (HSCAPP.LPSetting)

Description	Data Format	User Program Access
HSCAPP.LPSetting	long word (32-bit INT)	read/write

The HSCAPP.LPSetting is the lower setpoint (in counts) that defines when the HSC sub-system generates an interrupt.

The data loaded into the low preset must be greater than or equal to the data resident in the underflow (HSCAPP.UFSetting) parameter, or an HSC error is generated. (If the underflow and low preset values are negative numbers, the low preset must be a number with a smaller absolute value.)

Overflow Setting (HSCAPP.OFSetting)

Description	Data Format	Type	User Program Access
HSCAPP.OFSetting	long word (32-bit INT)	control	read/write

The HSCAPP.OFSetting defines the upper count limit for the counter. If the counter's accumulated value increments past the value specified in this variable, an overflow interrupt is generated. When the overflow interrupt is generated, the HSC sub-system rolls the accumulator over to the underflow value and the counter continues counting from the underflow value (counts are not lost in this transition). The user can specify any value for the overflow position, provided it is greater than the underflow value and falls between -2,147,483,648 and 2,147,483,647.

TIP Data loaded into the overflow variable must be greater than or equal to the data resident in the high preset (HSCAPP.HPSetting) or an HSC error is generated.

Underflow Setting (HSCAPP.UFSetting)

Description	Data Format	User Program Access
HSCAPP.UFSetting	long word (32-bit INT)	read/write

The HSCAPP.UFSetting defines the lower count limit for the counter. If the counter's accumulated value decrements past the value specified in this variable, an underflow interrupt is generated. When the underflow interrupt is generated, the HSC sub-system resets the accumulated value to the overflow value and the counter then begins counting from the overflow value (counts are not lost in this transition). The user can specify any value for the underflow position, provided it is less than the overflow value and falls between -2,147,483,648 and 2,147,483,647.

TIP Data loaded into the underflow variable must be less than or equal to the data resident in the low preset (HSCAPP.LPSetting) or an HSC error is generated.

Output Mask Bits (HSCAPP.OutputMask)

Description	Data Format	User Program Access
HSCAPP.OutputMask	word (32-bit binary)	read/write

The HSCAPP.OutputMask defines which embedded outputs on the controller can be directly controlled by the high-speed counter. The HSC sub-system has the ability to directly (without control program interaction) turn outputs ON or OFF based on the HSC accumulator reaching the High or Low presets. The bit pattern stored in the HSCAPP.OutputMask variable defines which outputs are controlled by the HSC and which outputs are not controlled by the HSC.

For example, if the user wants to control outputs 0, 1, 3, using HSC then the user needs to assign,
HscAppData.OutputMask = 2#1011
(OR using Decimal Value: HscAppData.OutputMask = 11)

The bit pattern of the HSCAPP.OutputMask variable directly corresponds to the output bits on the controller. Bits that are set (1) are enabled and can be turned on or off by the HSC sub-system. Bits that are clear (0) cannot be turned on or off by the HSC sub-system. The mask bit pattern can be configured only during initial setup.

The following table shows example of how HPOutput and OutputMask controls Embedded output.

Effect of HSC Output Mask on Embedded Outputs

Output Variable	32-Bit Signed Integer Data Word																				
	32…20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSCAPP.HPOutput (high preset output)		0	1	0	1	0	1	0	1	0	0	1	1	0	0	0	1	1	0	0	1
HSCAPP.OutputMask (output mask)		1	1	0	0	0	0	0	0	0	1	1	0	0	0	1	1	0	0	1	1
Embedded output (10-point)																				**0**	**1**
Embedded output (16-point)																**0**	**1**			**0**	**1**
Embedded output (24-point)												**1**				**0**	**1**			**0**	**1**

Effect of HSC Output Mask on Embedded Outputs

Output Variable	32-Bit Signed Integer Data Word																				
	32…20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Embedded output (48-point)		0	1								0	1				0	1			0	1

The outputs shown in the black boxes are the outputs under the control of the HSC sub-system. The mask defines which outputs can be controlled. The high preset output or low preset output values (HSCAPP.HPOutput or HSCAPP.LPOutput) define if each output is either ON (1) or OFF (0). Another way to view this is that the high or low preset output is written through the output mask, with the output mask acting like a filter.

The bits in the gray boxes are unused. For the 10-point controller, the first 4 bits of the mask word are used and the remaining mask bits are not functional because they do not correlate to any physical outputs on the base unit. For the 16, 24 and 48-point controllers, the first 6, 10 and 20 bits of the mask word are used, respectively.

The mask bit pattern can be configured only during initial setup.

High Preset Output (HSCAPP.HPOutput)

Description	Data Format	User Program Access
HSCAPP.HPOutput	long word (32-bit binary)	read/write

The High Preset Output defines the state (1 = ON or 0 = OFF) of the outputs on the controller when the high preset is reached. For more information on how to directly turn outputs on or off based on the high preset being reached, see Output Mask Bits (HSCAPP.OutputMask) on page 142.

The high output bit pattern can be configured during initial setup, or while the controller is operating. Use the HSC function block to load the new parameters while the controller is operating.

Low Preset Output (HSCAPP.LPOutput)

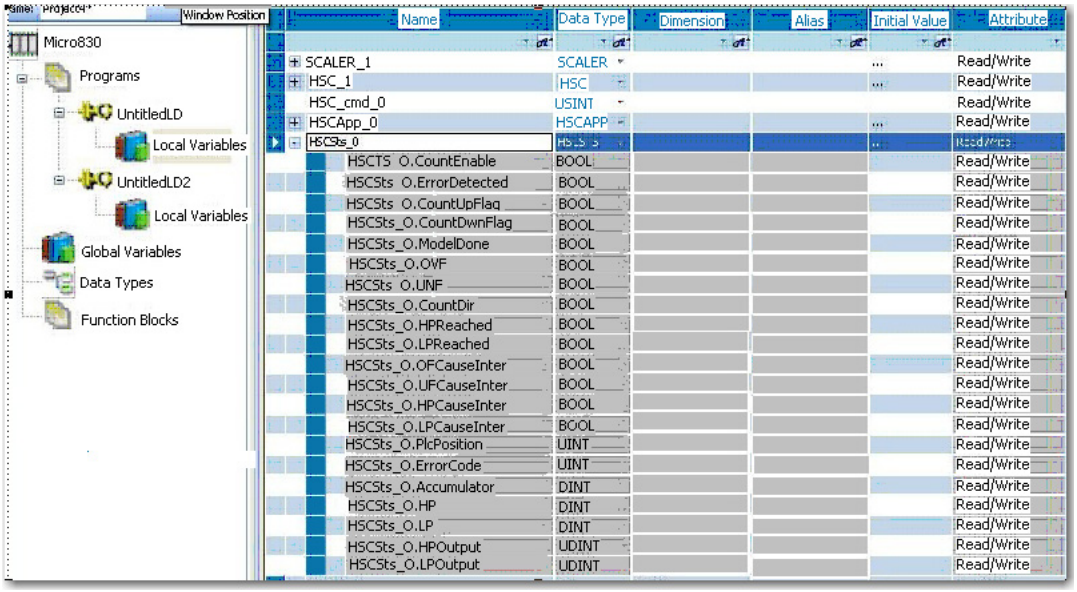
Description	Data Format	User Program Access
HSCAPP.LPOutput	long word (32-bit binary)	read/write

The Low Preset Output defines the state (1 = “on”, 0 = “off”) of the outputs on the controller when the low preset is reached. See Output Mask Bits (HSCAPP.OutputMask) on page 142 for more information on how to directly turn outputs on or off based on the low preset being reached.

The low output bit pattern can be configured during initial setup, or while the controller is operating. Use the HSC function block to load the new parameters while the controller is operating.

HSC STS (HSC Status) Data Structure

Define a HSC STS data (HSC status information data, data type HSCSTS) when programming a HSC.



Counting Enabled (HSCSTS.CountEnable)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.CountEnable	bit	0…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Counting Enabled control bit is used to indicate the status of the High-Speed Counter, whether counting is enabled (1) or disabled (0, default).

Error Detected (HSCSTS.ErrorDetected)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.ErrorDetected	bit	0…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Error Detected flag is a status bit that can be used in the control program to detect if an error is present in the HSC sub-system. The most common type of error that this bit represents is a configuration error. When this bit is set (1), you should look at the specific error code in parameter HSCSTS.ErrorCode. This bit is maintained by the controller and is set when there is an HSC error. This bit can be cleared by the user, if necessary.

Count Up (HSCSTS.CountUpFlag)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.CountUpFlag	bit	0…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Count Up bit is used with all of the HSCs (modes 0...9). If the HSCSTS.CountEnable bit is set, the Count Up bit is set (1). If the HSCSTS.CountEnable is cleared, the Count Up bit is cleared (0).

Count Down (HSCSTS.CountDownFlag)

Description	Data Format	HSC Modes(1)	User Program Access
SCSTS.CountDownFlag	bit	2…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Count Down bit is used with the bidirectional counters (modes 2...9). If the HSCSTS.CountEnable bit is set, the Count Down bit is set (1). If the HSCSTS.CountEnable bit is clear, the Count Down bit is cleared (0).

Mode Done (HSCSTS.Mode1Done)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.Mode1Done	bit	0 or 1	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Mode Done status flag is set (1) by the HSC sub-system when the HSC is configured for Mode 0 or Mode 1 behavior, and the accumulator counts up to the High Preset.

Overflow (HSCSTS.OVF)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.OVF	bit	0…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The HSCSTS.OVF status flag is set (1) by the HSC sub-system whenever the accumulated value (HSCSTS.Accumulator) has counted through the overflow variable (HSCAPP.OFSetting).

This bit is transitional and is set by the HSC sub-system. It is up to the control program to utilize, track if necessary, and clear (0) the overflow condition.

Overflow conditions do not generate a controller fault.

Underflow (HSCSTS.UNF)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.UNF	bit	0…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Underflow status flag is set (1) by the HSC sub-system whenever the accumulated value (HSCSTS.Accumulator) has counted through the underflow variable (HSCAPP.UFSetting).

This bit is transitional and is set by the HSC sub-system. It is up to the control program to utilize, track if necessary, and clear (0) the underflow condition.

Underflow conditions do not generate a controller fault.

Count Direction (HSCSTS.CountDir)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.CountDir	bit	0…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Count Direction status flag is controlled by the HSC sub-system. When the HSC accumulator counts up, the direction flag is set (1). Whenever the HSC accumulator counts down, the direction flag is cleared (0).

If the accumulated value stops, the direction bit retains its value. The only time the direction flag changes is when the accumulated count reverses.

This bit is updated continuously by the HSC sub-system whenever the controller is in a run mode.

High Preset Reached (HSCSTS.HPReached)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.HPReached	bit	2…9	read/write

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The High Preset Reached status flag is set (1) by the HSC sub-system whenever the accumulated value (HSCSTS.Accumulator) is greater than or equal to the high preset variable (HSCAPP.HPSetting).

This bit is updated continuously by the HSC sub-system whenever the controller is in an executing mode. Writing to this element is not recommended.

Low Preset Reached (HSCSTS.LPReached)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.LPReached)	bit	2…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Low Preset Reached status flag is set (1) by the HSC sub-system whenever the accumulated value (HSCSTS.Accumulator is less than or equal to the low preset variable HSCAPP.LPSetting).

This bit is updated continuously by the HSC sub-system whenever the controller is in an executing mode. Writing to this element is not recommended.

Overflow Interrupt (HSCSTS.OFCauseInter)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.OFCauseInter	bit	0…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Overflow Interrupt status bit is set (1) when the HSC accumulator counts through the overflow value and the HSC interrupt is triggered. This bit can be used in the control program to identify that the overflow variable caused the HSC interrupt. If the control program needs to perform any specific control action based on the overflow, this bit is used as conditional logic.

This bit can be cleared (0) by the control program and is also cleared by the HSC sub-system whenever these conditions are detected:

- Low Preset Interrupt executes
- High Preset Interrupt executes
- Underflow Interrupt executes

Underflow Interrupt (HSCSTS.UFCauseInter)

Description	Data Format	HSC Modes(1)	User Program Access
HSCSTS.UFCauseInter	bit	2…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Underflow Interrupt status bit is set (1) when the HSC accumulator counts through the underflow value and the HSC interrupt is triggered. This bit can be used in the control program to identify that the underflow condition caused the HSC interrupt. If the control program needs to perform any specific control action based on the underflow, this bit is used as conditional logic.

This bit can be cleared (0) by the control program and is also cleared by the HSC sub-system whenever these conditions are detected:

- Low Preset Interrupt occurs

- High Preset Interrupt occurs
- Overflow Interrupt occurs

High Preset Interrupt (HSCSTS.HPCauseInter)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.HPCauseInter	bit	0…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The High Preset Interrupt status bit is set (1) when the HSC accumulator reaches the high preset value and the HSC interrupt is triggered. This bit can be used in the control program to identify that the high preset condition caused the HSC interrupt. If the control program needs to perform any specific control action based on the high preset, this bit is used as conditional logic.

This bit can be cleared (0) by the control program and is also cleared by the HSC sub-system whenever these conditions are detected:

- Low Preset Interrupt occurs
- Underflow Interrupt occurs
- Overflow Interrupt occurs

Low Preset Interrupt (HSCSTS.LPCauseInter)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.LPCauseInter	bit	2…9	read/write

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Low Preset Interrupt status bit is set (1) when the HSC accumulator reaches the low preset value and the HSC interrupt has been triggered. This bit can be used in the control program to identify that the low preset condition caused the HSC interrupt. If the control program needs to perform any specific control action based on the low preset, this bit would be used as conditional logic.

This bit can be cleared (0) by the control program and is also be cleared by the HSC sub-system whenever these conditions are detected:

- High Preset Interrupt occurs
- Underflow Interrupt occurs
- Overflow Interrupt occurs

Programmable Limit Switch Position (HSCSTS.PLSPosition)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.PLSPosition	Word (INT)	0…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

When the HSC is in Counting mode, and PLS is enabled, this parameter indicates which PLS element is used for the current HSC configuration.

Error Code (HSCSTS.ErrorCode)

Description	Data Format	HSC Modes[1]	User Program Access
HSCSTS.ErrorCode	Word (INT)	0…9	read only

(1) For Mode descriptions, see HSC Mode (HSCAPP.HSCMode) on page 134.

The Error Codes detected by the HSC sub-system are displayed in this word. Errors include:

Error Code Sub-element	HSC counting Error Code	Error Description
Bit 15…8 (high byte)	0…255	The non-zero value for high byte indicates that the HSC error is due to PLS data setting. The value of high byte indicates which element of PLS data triggers the error.
Bit 7-0 (low byte)	0x00	No error
	0x01	Invalid HSC counting mode
	0x02	Invalid High preset
	0x03	Invalid overflow
	0x04	Invalid underflow
	0x05	No PLS data

Writing to this element is not recommended except for clearing existing errors and to capture new HSC errors.

Accumulator (HSCSTS.Accumulator)

Description	Data Format	User Program Access
HSCSTS.Accumulator	long word (32-bit INT)	read only

HSCSTS.Accumulator contains the number of counts detected by the HSC sub-system. If either mode 0 or mode 1 is configured, the accumulator is reset to 0 when a high preset is reached or when an overflow condition is detected.

High Preset (HSCSTS.HP)

Description	Data Format	User Program Access
HSCSTS.HP	long word (32-bit INT)	read only

The HSCSTS.HP is the upper setpoint (in counts) that defines when the HSC sub-system generates an interrupt.

The data loaded into the high preset must be less than or equal to the data resident in the overflow (HSCAPP.OFSetting) parameter or an HSC error is generated.

This is the latest high preset setting, which may be updated by PLS function from the PLS data block.

Low Preset (HSCSTS.LP)

Description	Data Format	User Program Access
HSCSTS.LP	long word (32-bit INT)	read only

The HSCSTS.LP is the lower setpoint (in counts) that defines when the HSC sub-system generates an interrupt.

The data loaded into the low preset must greater than or equal to the data resident in the underflow (HSCAPP.UFSetting) parameter, or an HSC error is generated. If the underflow and low preset values are negative numbers, the low preset must be a number with a smaller absolute value.

This is the latest low preset setting, which may be updated by PLS function from the PLS data block.

High Preset Output (HSCSTS.HPOutput)

Description	Data Format	User Program Access
HSCSTS.HPOutput	long word (32-bit binary)	read only

The High Preset Output defines the state (1 = ON or 0 = OFF) of the outputs on the controller when the high preset is reached. See Output Mask Bits (HSCAPP.OutputMask) on page 142 for more information on how to directly turn outputs on or off based on the high preset being reached.

This is the latest high preset output setting, which may be updated by PLS function from the PLS data block.

Low Preset Output (HSCSTS.LPOutput)

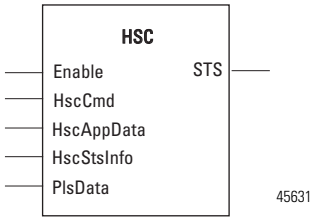
Description	Data Format	User Program Access
HSCSTS.LPOutput	long word (32-bit binary))	read only

The Low Preset Output defines the state (1 = “on”, 0 = “off”) of the outputs on the controller when the low preset is reached. See Output Mask Bits (HSCAPP.OutputMask) on page 142 for more information on how to directly turn outputs on or off based on the low preset being reached.

This is the latest low preset output setting, which may be updated by PLS function from the PLS data block.

HSC (High Speed Counter) Function Block

The HSC function block can be used to start/stop HSC counting, to refresh HSC status, to reload HSC setting, and to reset HSC accumulator.



HSC Parameters

Parameter	Parameter Type	Data Type	Parameter Description
Enable	Input	BOOL	Enable function block. When Enable = TRUE, perform the HSC operation specified in "HSC command" parameter. When Enable = FALSE, there is no HSC operation, and no HSC status update.
HscCmd	Input	USINT	Refer to HSC Commands on page 152
HscAppData	Input	See HSC APP Data Structure on page 133	HSC application configuration. Only initial configuration is needed usually.
PlsData	Input	See array of Programmable Limit Switch (PLS) Function on page 153	Programmable Limit Switch (PLS) Data
HscStsInfo	Output	See HSC STS (HSC Status) Data Structure on page 144	HSC dynamic status. Status info is usualy continuously updated during HSC counting.
Sts	Output	UINT	HSC function block execution status

HSC Commands (HScCmd)

HscCmd is an input parameter with data type USINT. All HSC commands (1...4) are Level commands. Users are advised to disable the instruction before updating the command.

HscCmd = 1 starts the HSC mechanism. Once the HSC is in running mode, the **HscCmd = 2** must be issued to stop counting. Setting the Enable input parameter to False does not stop counting while in running mode.

HscCmd = 3 reloads the following parameter values: HighPreset, LowPreset, OverFlow, UnderFlow, HighPreset Output, and LowPreset Output.

The parameter values shown in the Variable Monitor may not match the values in the Hardware. Command 3 must be executed to load the values from the variables to the hardware without stopping the HSC.

If the HSC Enable is True, HscCmd = 3 will continuously load the parameters. Trigger HscCmd = 3 only once.

HscCmd = 4 (reset) sets the Acc value to the HSC AppData.Accumalator value. The HscCmd =4 does not stop HSC counting. If HSC is counting when the HscCmd =4 is issued, some counting may be lost.

To reset the Acc value and then continue the counting, trigger the HscCmd =4 only once. If the command is enabled continuously, it may cause errors.

HSC AppData.Accumalator value is updated automatically by the HSC mechanism with the same value as the HSC Sts.Accumulator. To set one specific value to HSC Acc while counting, write the value to HSC AppData.Accumalator immediately before HscCmd =4 is issued.

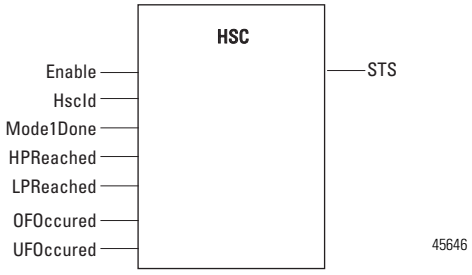
HSC Commands

HSC Command	Description
0x00	Reserved
0x01	HSC RUN • Start HSC (if HSC in Idle mode and Rung is Enabled) • Update HSC Status Info only (if HSC already in RUN mode and Rung is Enabled) • Update HSC status Info only (if Rung is disabled)
0x02	HSC Stop: Stop a HSC counting (if HSC is in RUN mode and Rung is Enabled.)
0x03	HSC Load: reload HSC Configuration (if Rung is Enabled) for 6 input elements: HPSetting, LPSetting, HPOutput, LPOutput, OFSetting, and UFSetting. HSC accumulator is NOT reloaded by cmd = 0x03.
0x04	HSC Reset: set Accumulator to assigned value, and reset HSC status information (if Rung is Enabled)

HSC Function Block Status Codes

HSC Status Code	Description
0x00	No action from Controller because the function block is not enabled
0x01	HSC function block successfully executed
0x02	HSC command invalid
0x03	HSC ID out of range
0x04	HSC Configuration Error

HSC_SET_STS Function Block



The HSC Set Status function block can be used to change the HSC counting status. This function block is called when the HSC is not counting (stopped).

HSC Parameters

Parameter	Parameter Type	Data Type	Parameter Description
Enable	Input	BOOL	Enable function block. When Enable = TRUE, set/reset the HSC status. When Enable = FALSE, there is no HSC status change.
HscId	Input	See HSC APP Data Structure on page 133	Describes which HSC status to set.
Mode1Done	Input	BOOL	Mode 1A or 1B counting is done.
HPReached	Input	BOOL	High Preset reached. This bit can be reset to FALSE when HSC is not counting.
LPReached	Input	BOOL	Low Preset reached. This bit can be reset to FALSE when HSC is not counting.
OFOccurred	Input	BOOL	Overflow occurred. This bit can be reset to FALSE when necessary.
UFOccurred	Input	BOOL	Underflow occurred. This bit can be reset to FALSE when necessary.
Sts	Output	UINT	HSC function block execution status Refer to HSC Function Block Status Codes on page 152 for HSC status code description (except 0x02 and 0x04).

Programmable Limit Switch (PLS) Function

The Programmable Limit Switch function allows you to configure the High-Speed Counter to operate as a PLS (programmable limit switch) or rotary cam switch.

When PLS operation is enabled (HSCAPP.PLSEnable = True), the HSC (High-Speed Counter) uses PLS data for limit/cam positions. Each limit/cam position has corresponding data parameters that are used to set or clear physical outputs on the controller's base unit. The PLS data block is illustrated below.

IMPORTANT The PLS Function only operates in tandem with the HSC of a Micro830 controller. To use the PLS function, an HSC must first be configured.

PLS Data structure

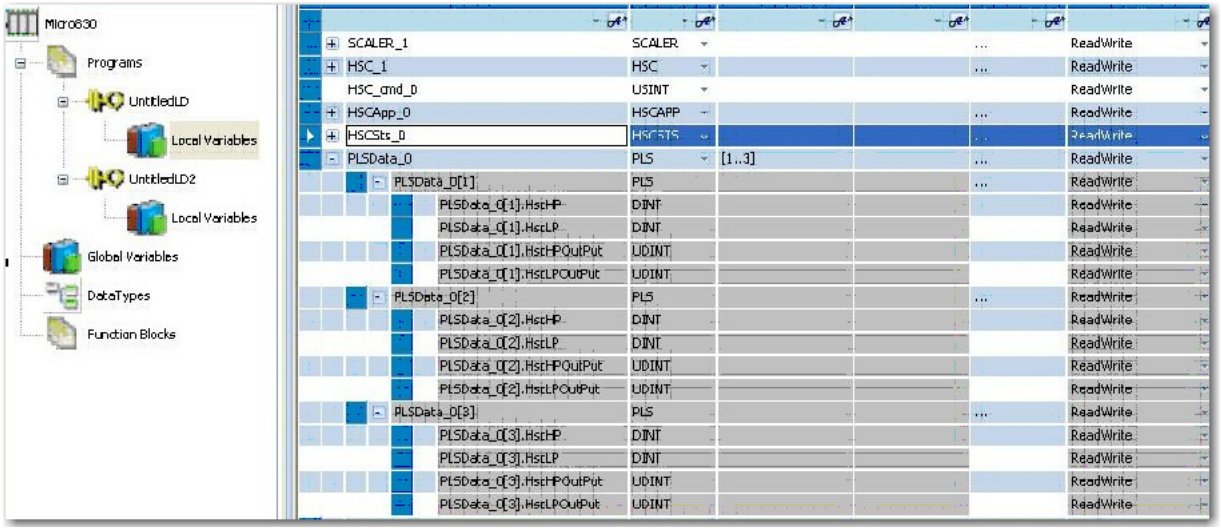
The Programmable Limit Switch function is an additional set of operating modes for the High Speed Counter. When operating in these modes, the preset and output data values are updated using user supplied data each time one of the presets is reached. These modes are programmed by providing a PLS data block that contains the data sets to be used.

PLS data structure is a flexible array, with each element defined as follows,

Element Order	Data Type	Element Description
Word 0…1	DINT	High preset setting
Word 2…3	DINT	Low preset setting
Word 4…5	UDINT	High preset Output data
Word 6…7	UDINT	Low preset Output data

The total number of elements for one PLS data cannot be larger than 255.

When PLS is not enabled, PLS data are still required to be defined, but can be not initialized.



PLS Operation

When the PLS function is enabled, and the controller is in the run mode, the HSC counts incoming pulses. When the count reaches the first preset (HSCHP or HSCLP) defined in the PLS data, the output source data (HSCHPOutput or HSCLPOutput) is written through the HSC mask (HSCAPP.OutputMask).

At that point, the next presets (HSCHP and HSCLP) defined in the PLS data become active.

When the HSC counts to that new preset, the new output data is written through the HSC mask. This process continues until the last element within the PLS data block is loaded. At that point the active element within the PLS data block is reset to zero. This behavior is referred to as circular operation.

TIP The HSCHPOutput is only written when HSCHP is reached. The HSCLPOutput is written when HSCLP is reached.

TIP Output High Data is only operational when the counter is counting up. Output Low Data is only operational when the counter is counting down.

If invalid data is loaded during operation, an HSC error is generated and causes a controller fault.

You can use the PLS in Up (high), Down (low), or both directions. If your application only counts in one direction, ignore the other parameters.

The PLS function can operate with all of the other HSC capabilities. The ability to select which HSC events generate a user interrupt are not limited.

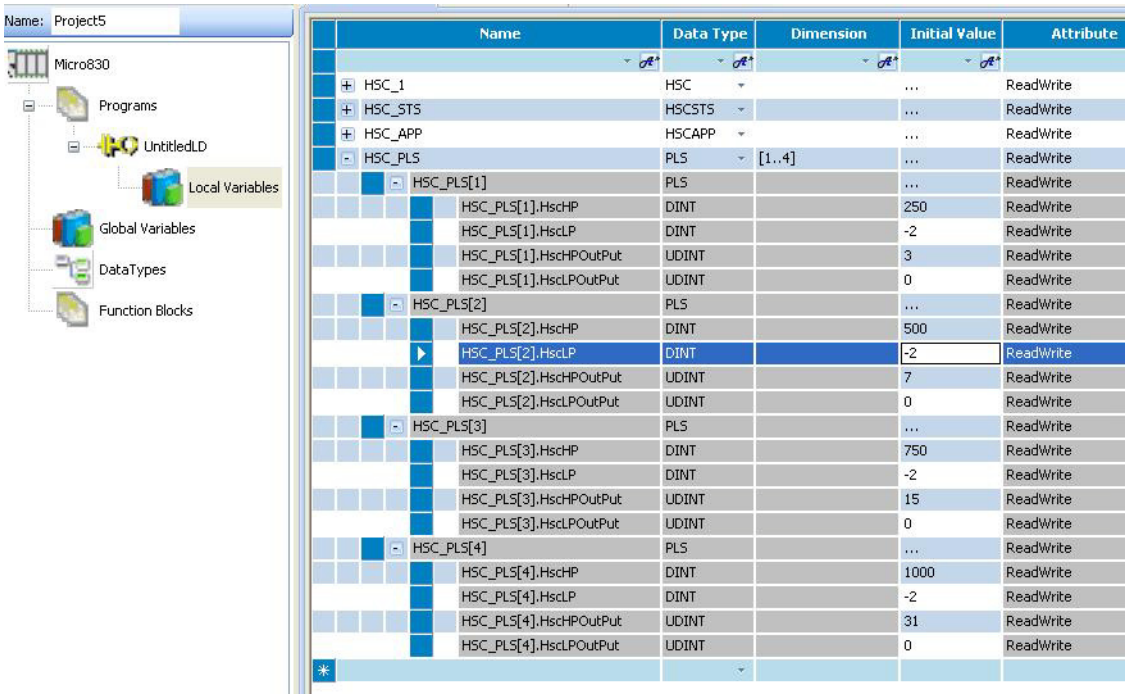
PLS Example

Setting Up the PLS data

Using Connected Components Workbench, define the PLS data HSC_PLS's dimension as [1..4].

PLS Data Definition

Data	Description	Data Format
HSCHP	High Preset	32-bit signed integer
HSCLP	Low Preset	
HSCHPOutput	Output High Data	32-bit binary (bit 31--> 0000 0000 0000 0000 0000 0000 0000 0000 <--bit 0)
HSCLPOutput	Output Low Data	



Name	Data Type	Dimension	Initial Value	Attribute
HSC_1	HSC		...	ReadWrite
HSC_STS	HSCSTS		...	ReadWrite
HSC_APP	HSCAPP		...	ReadWrite
HSC_PLS	PLS	[1..4]	...	ReadWrite
HSC_PLS[1]	PLS		...	ReadWrite
HSC_PLS[1].HscHP	DINT		250	ReadWrite
HSC_PLS[1].HscLP	DINT		-2	ReadWrite
HSC_PLS[1].HscHPOutPut	UDINT		3	ReadWrite
HSC_PLS[1].HscLPOutPut	UDINT		0	ReadWrite
HSC_PLS[2]	PLS		...	ReadWrite
HSC_PLS[2].HscHP	DINT		500	ReadWrite
HSC_PLS[2].HscLP	DINT		-2	ReadWrite
HSC_PLS[2].HscHPOutPut	UDINT		7	ReadWrite
HSC_PLS[2].HscLPOutPut	UDINT		0	ReadWrite
HSC_PLS[3]	PLS		...	ReadWrite
HSC_PLS[3].HscHP	DINT		750	ReadWrite
HSC_PLS[3].HscLP	DINT		-2	ReadWrite
HSC_PLS[3].HscHPOutPut	UDINT		15	ReadWrite
HSC_PLS[3].HscLPOutPut	UDINT		0	ReadWrite
HSC_PLS[4]	PLS		...	ReadWrite
HSC_PLS[4].HscHP	DINT		1000	ReadWrite
HSC_PLS[4].HscLP	DINT		-2	ReadWrite
HSC_PLS[4].HscHPOutPut	UDINT		31	ReadWrite
HSC_PLS[4].HscLPOutPut	UDINT		0	ReadWrite

Once the values above for all 4 PLS data elements have been entered, the PLS is configured.

Assume that HSCAPP.OutputMask = 31 (HSC mechanism controls Embedded Output 0...4 only), and HSCAPP.HSCMode = 0.

PLS Operation for This Example

When the ladder logic first runs, HSCSTS.Accumulator = 1, therefore all the outputs are turned off. The value of HSCSTS.HP = 250

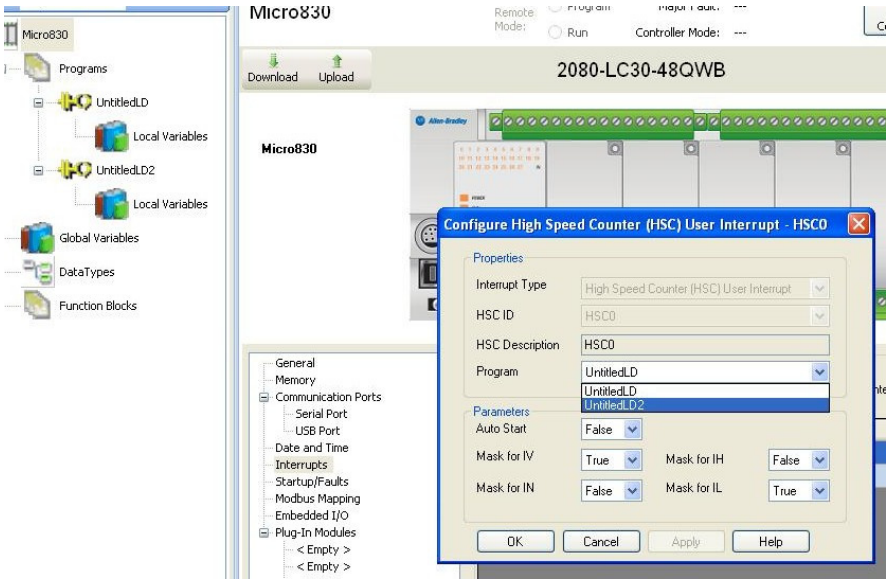
When HSCSTS.Accumulator = 250, the HSC_PLS[1].HscHPOutput is sent through the HSCAPP.OutputMask and energizes the outputs 0 and 1.

This will repeat as the HSCSTS.Accumulator reaches 500, 750, and 1000. The controller energizes outputs 0...2, 0...3, and 0...4 respectively. Once completed, the cycle resets and repeats from HSCSTS.HP = 250.

HSC Interrupts

An interrupt is an event that causes the controller to suspend the task it is currently performing, perform a different task, and then return to the suspended task at the point where it suspended. Micro800 supports up to six HSC interrupts.

An HSC interrupt is a mechanism that Micro830 and Micro850 controllers provide to execute selected user logic at a pre-configured event.

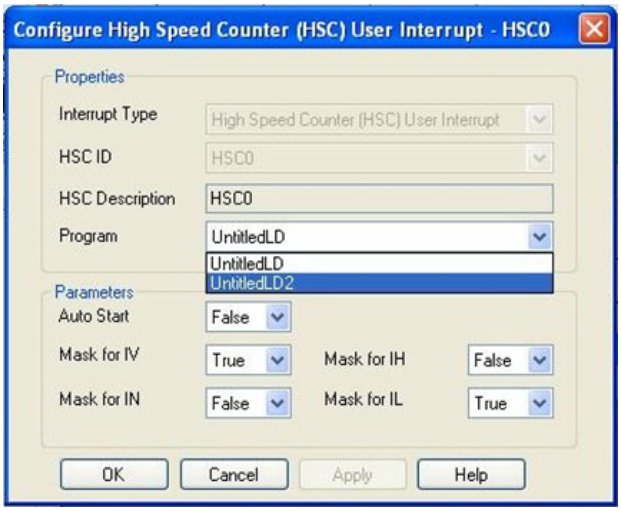


HSC0 is used in this document to define how HSC interrupts work.

HSC Interrupt Configuration

In the User Interrupt configuration window, select HSC, and HSC ID, which is the interrupt triggering the User Interrupt.

The following diagram shows the selectable fields in the Interrupt configuration window.



HSC Interrupt POU

This is the name of the Program Organizational Unit (POU) which is executed immediately when this HSC Interrupt occurs. You can choose any pre-programmed POU from the drop-down list.

Auto Start (HSC0.AS)

Description	Data Format	HSC Modes[1]	User Program Access
AS - Auto Start	bit	0…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The Auto Start is configured with the programming device and stored as part of the user program. The auto start bit defines if the HSC interrupt function automatically starts whenever the controller enters any run or test mode.

Mask for IV (HSC0.MV)

Description	Data Format	HSC Modes[1]	User Program Access
MV - Overflow Mask	bit	0…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145..

The MV (Overflow Mask) control bit is used to enable (allow) or disable (not allow) an overflow interrupt from occurring. If this bit is clear (0), and an overflow reached condition is detected by the HSC, the HSC user interrupt is not executed.

This bit is controlled by the user program and retains its value through a power cycle. It is up to the user program to set and clear this bit.

Mask for IN (HSC0.MN)

Description	Data Format	HSC Modes[1]	User Program Access
MN - Underflow Mask	bit	2…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The MN (Underflow Mask) control bit is used to enable (allow) or disable (not allow) a underflow interrupt from occurring. If this bit is clear (0), and a Underflow Reached condition is detected by the HSC, the HSC user interrupt is not executed.

This bit is controlled by the user program and retains its value through a power cycle. It is up to the user program to set and clear this bit.

Mask for IH (HSC0.MH)

Description	Data Format	HSC Modes(1)	User Program Access
MH - High Preset Mask	bit	0…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The MH (High Preset Mask) control bit is used to enable (allow) or disable (not allow) a high preset interrupt from occurring. If this bit is clear (0), and a High Preset Reached condition is detected by the HSC, the HSC user interrupt is not executed.

This bit is controlled by the user program and retains its value through a power cycle. It is up to the user program to set and clear this bit.

Mask for IL (HSC0.ML)

Description	Data Format	HSC Modes(1)	User Program Access
ML - Low Preset Mask	bit	2…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The ML (Low Preset Mask) control bit is used to enable (allow) or disable (not allow) a low preset interrupt from occurring. If this bit is clear (0), and a Low Preset Reached condition is detected by the HSC, the HSC user interrupt is not executed.

This bit is controlled by the user program and retains its value through a power cycle. It is up to the user program to set and clear this bit.

HSC Interrupt Status Information

User Interrupt Enable (HSC0.Enabled)

Description	Data Format	HSC Modes(1)	User Program Access
HSC0.Enabled	bit	0…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The Enabled bit is used to indicate HSC interrupt enable or disable status.

User Interrupt Executing (HSC0.EX)

Description	Data Format	HSC Modes(1)	User Program Access
HSC0.EX	bit	0…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The EX (User Interrupt Executing) bit is set (1) whenever the HSC sub-system begins processing the HSC subroutine due to any of the following conditions:

- Low preset reached
- High preset reached
- Overflow condition – count up through the overflow value
- Underflow condition – count down through the underflow value

The HSC EX bit can be used in the control program as conditional logic to detect if an HSC interrupt is executing.

The HSC sub-system will clear (0) the EX bit when the controller completes its processing of the HSC subroutine.

User Interrupt Pending (HSC0.PE)

Description	Data Format	HSC Modes[1]	User Program Access
HSC0.PE	bit	0…9	read only

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The PE (User Interrupt Pending) is a status flag that represents an interrupt is pending. This status bit can be monitored or used for logic purposes in the control program if you need to determine when a subroutine cannot be executed immediately. This bit is maintained by the controller and is set and cleared automatically.

User Interrupt Lost (HSC0.LS)

Description	Data Format	HSC Modes[1]	User Program Access
HSC0.LS	bit	0…9	read/write

(1) For Mode descriptions, see Count Down (HSCSTS.CountDownFlag) on page 145.

The LS (User Interrupt Lost) is a status flag that represents an interrupt has been lost. The controller can process 1 active and maintain up to 1 pending user interrupt conditions before it sets the lost bit.

This bit is set by the controller. It is up to the control program to utilize, track the lost condition if necessary.

Use HSC

To use HSC, refer to Use the High Speed Counter on page 214.

Controller Security

Micro800 security generally has two components:

- **Exclusive Access** which prevents simultaneous configuration of the controller by two users
- **Controller Password Protection** which secures the Intellectual Property contained within the controller and prevents unauthorized access

Exclusive Access

Exclusive access is enforced on the Micro800 controller regardless of whether the controller is password-protected or not. This means that only one Connected Components Workbench session is authorized at one time and only an authorized client has exclusive access to the controller application. This ensures that only one software session has exclusive access to the Micro800 application-specific configuration.

Exclusive access is enforced on Micro800 firmware revision 1 and 2. When a Connected Components Workbench user connects to a Micro800 controller, the controller is given exclusive access to that controller.

Password Protection

By setting a password on the controller, a user effectively restricts access to the programming software connections to the controller to software sessions that can supply the correct password. Essentially, Connected Components Workbench operation such as upload and download are prevented if the controller is secured with a password and the correct password is not provided.

Micro800 controllers with firmware revision 2 and later are shipped with no password but a password can be set through the Connected Components Workbench software (revision 2 or later).

In Connected Components Workbench software revision 10 or later, a stronger password algorithm is introduced to provide better security. To take full advantage of this enhancement, the Micro800 controller must have firmware revision 10 or later, and the project must also be version 10 or later.

The controller password is also backed up to the memory backup module (that is, 2080-MEMBAK-RTC for Micro830 and Micro850; 2080-LCD for Micro810; and microSD card for Micro820 controllers).

TIP For instructions on how to set, change, and clear controller passwords, see Configure Controller Password on page 210.

Compatibility

The Controller Password feature is supported on:

- Connected Components Workbench **revision 2** and later
- Micro800 controllers with **revision 2** firmware

For users with earlier versions of the software and/or hardware, refer to the compatibility scenarios below.

Connected Components Workbench revision 1 with Micro800 controller firmware revision 2

Connection to a Micro800 controller with firmware revision 2 using an earlier version of the Connected Components Workbench software (revision 1) is possible and connections will be successful. However, the software will not be able to determine whether the controller is locked or not.

If the controller is not locked, access to the user application will be allowed, provided the controller is not busy with another session. If the controller is locked, access to the user application will fail. Users will need to upgrade to revision 2 of the Connected Components Workbench software.

Connected Components Workbench revision 2 with Micro800 controller firmware revision 1

Connected Components Workbench revision 2 is capable of "discovering" and connecting to Micro800 controllers with firmware revision earlier than revision 2 (that is, not supporting the Controller Password feature). However, the Controller Password feature will not be available to these controllers. The user will not be able see interfaces associated with the Controller Password feature in the Connected Components Workbench session.

Users are advised to upgrade the firmware. See Flash Upgrade Your Micro800 Firmware on page 201 for instructions.

ATTENTION: Connected Components Workbench revision 9 or earlier with Micro800 controller revision 10 or later

If a Micro800 controller with firmware revision 10 or later is locked using the new password algorithm introduced in Connected Components Workbench software revision10 or later, it cannot be accessed using Components Workbench software revision 9 or earlier. Users are advised to upgrade to the latest version of Connected Components Workbench software.

Work with a Locked Controller

The following workflows are supported on compatible Micro800 controllers (firmware revision 2) and Connected Components Workbench software revision 2.

Upload from a Password-Protected Controller

1. Launch the Connected Components Workbench software.
2. In the Project Organizer, expand Catalog by clicking the + sign.
3. Select the target controller.
4. Select Upload.
5. When requested, provide the controller password.

IMPORTANT When using Connected Components Workbench software revision 9 or earlier:

- You cannot upload a revision 10 or later project from the controller.
- You can upload a revision 9 or earlier project from the controller if it was downloaded to the controller using Connected Components Workbench software revision 10 or later, but you cannot go online.

Debug a Password-Protected Controller

To debug a locked controller, you have to connect to the controller through the Connected Components Workbench software and provide the password before you can proceed to debug.

1. Launch the Connected Components Workbench software.
2. In the Project Organizer, expand Catalog by clicking the + sign.
3. Select the catalog number of your controller.
4. When requested, provide the controller password.
5. Build and save your project.
6. Debug.

Download to a Password-Protected Controller

1. Launch the Connected Components Workbench software.
2. Click Connect.
3. Select the target controller.
4. When requested, provide the controller password.
5. Build and save the project, if needed.

6. Click Download.
7. Click Disconnect.

IMPORTANT If the controller has a password locked revision 10 or later project, you cannot access the controller using Connected Workbench software revision 9 or earlier. If you use Connected Components Workbench software revision 10 or later to download a revision 9 or earlier project, the password in the controller will be automatically converted to the old algorithm.

IMPORTANT If the controller has a password locked revision 9 or earlier project and you use Connected Components Workbench software revision 10 or later, to download a revision 10 or later project, the password in the controller will be automatically converted to the new algorithm.

IMPORTANT If communication is lost during the download, repeat the download and verify that the controller is password protected.

Transfer Controller Program and Password-Protect Receiving Controller

In this scenario, the user needs to transfer user application from controller1 (locked) to another Micro800 controller with the same catalog number. The transfer of the user application is done through the Connected Components Workbench software by uploading from controller1, then changing the target controller in the Micro800 project, and then downloading to controller2. Finally, controller2 will be locked.

1. In the Project Organizer, click the Discover icon.
   The Browse Connections dialog appears.
2. Select target controller1.
3. When requested, enter the controller password for controller1.
4. Build and save the project.
5. Click Disconnect.
6. Power down controller1.
7. Swap controller1 hardware with controller2 hardware.
8. Power up controller2.
9. Click Connect.
10. Select target controller2.
11. Click Download.
12. Lock controller2. See Configure Controller Password on page 210.

Back Up and Restore a Password-Protected Controller

In this workflow, user application will be backed up from a Micro800 controller that is locked to a memory plug-in device.

1. In the Project Organizer, click the Discover icon.
   The Browse Connections dialog appears.
2. Select the target controller.
3. When requested, enter the controller password.
4. Back up controller contents to the memory module.
   The project in the memory module is now password locked.
5. Remove the memory module from controller1 and insert into controller2.
6. Restore contents from the memory module to controller2.
   This operation succeeds only if:
   - the controller has no password – the project can be restored to the controller by setting the "Load on power up" option for the memory module to Load Always.
   - the controller's password matches the project's password.

IMPORTANT	Even though the password matches, the restore operation will fail if either one of the controller or project in the memory module is protected using the old password algorithm, and the other is protected using the new password algorithm. You can flash update the controller using the Reset option to clear the password before restoring the project to the controller.

Configure Controller Password

To set, change, and clear controller password, see the quickstart instructions Configure Controller Password on page 210.

IMPORTANT	After creating or changing the controller password, you need to power down the controller in order for the password to be saved.

Recover from a Lost Password

If the controller is secured with a password and the password has been lost, then it becomes impossible to access the controller using the Connected Components Workbench software.

To recover, the controller must be set to Program Mode using the keyswitch for Micro830 and Micro850 controllers, the 2080-LCD for Micro810 controllers, or the 2080-REMLCD for Micro820 controllers. Then, ControlFlash can be used to update the controller firmware, which also clears the controller memory. In Connected Components Workbench software version 10 or later, the Reset option must be selected for the controller memory to be cleared during the firmware update. If the Upgrade or Downgrade option is selected, the password is retained.

ATTENTION: The project in the controller will be lost but a new project can be downloaded.

Specifications

IMPORTANT Specifications for the analog and discrete Micro800 plug-in and expansion I/O modules are available in the following Rockwell Automation publications:

- Micro800 Discrete and Analog Expansion I/O User Manual, publication 2080-UM003
- Micro800 Plug-in Modules User Manual, publication 2080-UM004

Micro830 Controllers

Micro830 10-Point Controllers

General – 2080-LC30-10QWB, 2080-LC30-10QVB

Attribute	2080-LC30-10QWB	2080-LC30-10QVB
Number of I/O	10 (6 inputs, 4 outputs)	
Dimensions HxWxD	90 x 100 x 80 mm (3.54 x 3.94 x 3.15 in.)	
Shipping weight, approx.	0.302 kg (0.666 lb)	
Wire size	0.14…2.5 mm^2 (26…14 AWG) solid copper wire or 0.14…1.5 mm^2 (26…14 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max	
Wiring category(1)	2 – on signal ports 2 – on power ports	
Wire type	Use copper conductors only	
Terminal screw torque	0.6 Nm (4.4 lb-in) max (using a 2.5 mm (0.10 in.) flat-blade screwdriver)	
Input circuit type	12/24V sink/source (standard) 24V sink/source (high-speed)	
Output circuit type	Relay	24V DC sink transistor (standard and high-speed)
Event input interrupt support	Yes	
Power consumption	7.88 W	
Power supply voltage range	20.4…26.4V DC Class 2	
I/O rating	Input 24V DC, 8.8 mA Output 2 A, 240V AC, general use	Input 24V DC, 8.8 mA Output 2 A, 24V DC, 1 A per point (Surrounding air temperature 30 °C) 24 V DC, 0.3 A per point (Surrounding air temperature 65 °C)
Isolation voltage	250V (continuous), Reinforced Insulation Type, Outputs to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720 V DC, Inputs to Aux and Network, 3250 V DC Outputs to Aux and Network, Inputs to Outputs	50V (continuous), Reinforced Insulation Type, I/O to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720 V DC, I/O to Aux and Network, Inputs to Outputs
Pilot duty rating	C300, R150	—

General – 2080-LC30-10QWB, 2080-LC30-10QVB

Attribute	2080-LC30-10QWB	2080-LC30-10QVB
Insulation stripping length	7 mm (0.28 in.)	
Enclosure type rating	Meets IP20	
North American temp code	T4	

(1) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

Inputs

Attribute	High-Speed DC Input (Inputs 0…3)	Standard DC Input (inputs 4 and higher)
Number of Inputs	4	2
Input group to backplane isolation	Verified by one of the following dielectric tests: 1,414V DC for 2 s 75V DC working voltage (IEC Class 2 reinforced insulation)	
Voltage category	24V DC sink/source	
Off-state voltage, max	5V DC	
On-state voltage , nom	24V DC	
On-state voltage range	16.8…26.4V DC @ 65 °C (149 °F) 16.8…30.0V DC @ 30 °C (86 °F)	10…26.4V DC @ 65 °C (149 °F) 10…30.0V DC @ 30 °C (86 °F)
Off-state current, max	1.5 mA	
On-state current, min	5.0 mA @ 16.8V DC	1.8 mA @ 10V DC
On-state current, nom	8.8 mA @ 24V DC	8.5 mA @ 24V DC
On-state current, max	12.0 mA @ 30V DC	
Nominal impedance	3 kΩ	3.74 kΩ
IEC input compatibility	Type 3	
AC input filter setting	8 ms for all embedded inputs (In Connected Components Workbench, go to the Embedded I/O configuration window to reconfigure the filter setting for each input group)	

Isolated AC Inputs (2080-LC30-10QWB, 2080-LC30-10QVB) (Inputs 0…3)

Attribute	Value
On-state voltage, nom	12/24V AC @ 50/60 Hz
Off-state voltage, min	4V AC @ 50/60Hz
Operating frequency, nom	50/60 Hz

Outputs

Attribute	2080-LC30-10QWB	2080-LC30-10QVB	
	Relay Output	Hi-Speed Output (Outputs 0…1)	Standard Output (Outputs 2…3)
Output voltage, min	5V DC, 5V AC	10.8V DC	10V DC
Output voltage, max	125V DC, 265V AC	26.4V DC	26.4V DC
Load current, min	10 mA	10 mA	
Load current, max	2.0 A	100 mA (high-speed operation) 1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)	1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)
Surge current, per point	Refer to Relay Contacts Ratings on page 169	4.0 A every 1 s @ 30 °C; every 2 s @ 65 °C(1)	
Current, per common, max	5 A	2 A	4 A
Current, per controller, max	1440V A	2 A	4 A
Turn on time/ Turn off time, max	10 ms	2..5 µs	0.1 ms 1.0 ms

(1) Applies for general purpose operation only. Does not apply for high-speed operation.

Relay Contacts Ratings

Maximum Volts	Amperes		Amperes Continuous	Volt-Amperes	
	Make	Break		Make	Break
120V AC	15 A	1.5 A	2.0 A	1800V A	180V A
240V AC	7.5 A	0.75 A			
24V DC	1.0 A		1.0 A	28V A	
125V DC	0.22 A				

Environmental Specifications

Attribute	Value
Temperature, operating	IEC 60068-2-1 (Test Ad, Operating Cold), IEC 60068-2-2 (Test Bd, Operating Dry Heat), IEC 60068-2-14 (Test Nb, Operating Thermal Shock): -20…65 °C (-4…149 °F)
Temperature, surrounding air, max	65 °C (149 °F)
Temperature, non-operating	IEC 60068-2-1 (Test Ab, Unpackaged Nonoperating Cold), IEC 60068-2-2 (Test Bb, Unpackaged Nonoperating Dry Heat), IEC 60068-2-14 (Test Na, Unpackaged Nonoperating Thermal Shock): -40…85 °C (-40…185 °F)
Relative humidity	IEC 60068-2-30 (Test Db, Unpackaged Damp Heat): 5…95% non-condensing
Vibration	IEC 60068-2-6 (Test Fc, Operating): 2 g @ 10…500 Hz

Environmental Specifications

Attribute	Value
Shock, operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): 25 g
Shock, non-operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): DIN mount: 25 g PANEL mount: 45 g
Emissions	CISPR 11 Group 1, Class A
ESD immunity	IEC 61000-4-2: 6 kV contact discharges 8 kV air discharges
Radiated RF immunity	IEC 61000-4-3: 10V/m with 1 kHz sine-wave 80% AM from 80…2000 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 900 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 1890 MHz 10V/m with 1 kHz sine-wave 80% AM from 2000…2700 MHz
EFT/B immunity	IEC 61000-4-4: ±2 kV at 5 kHz on power ports ±2 kV at 5 kHz on signal ports
Surge transient immunity	IEC 61000-4-5: ±1 kV line-line(DM) and ±2 kV line-earth(CM) on power ports ±1 kV line-line(DM) and ±2 kV line-earth(CM) on signal ports
Conducted RF immunity	IEC 61000-4-6: 10V rms with 1 kHz sine-wave 80% AM from 150 kHz…80 MHz

Certifications

Certification (when product is marked)(1)	Value
c-UL-us	UL Listed Industrial Control Equipment, certified for US and Canada. See UL File E322657. UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada. See UL File E334470.
CE	European Union 2004/108/EC EMC Directive, compliant with: EN 61326-1; Meas./Control/Lab., Industrial Requirements EN 61000-6-2; Industrial Immunity EN 61000-6-4; Industrial Emissions EN 61131-2; Programmable Controllers (Clause 8, Zone A & B) European Union 2006/95/EC LVD, compliant with: EN 61131-2; Programmable Controllers (Clause 11)
C-Tick	Australian Radiocommunications Act, compliant with: AS/NZS CISPR 11; Industrial Emissions

(1) See the Product Certification link at http://www.rockwellautomation.com/products/certification/ for Declaration of Conformity, Certificates, and other certification details.

Micro830 16-Point Controllers

General – 2080-LC30-16AWB, 2080-LC30-16QWB, 2080-LC30-16QVB

Attribute	2080-LC30-16AWB	2080-LC30-16QWB	2080-LC30-16QVB
Number of I/O	16 (10 inputs, 6 outputs)		
Dimensions HxWxD	90 x 100 x 80 mm (3.54 x 3.94 x 3.15 in.)		
Shipping weight, approx.	0.302 kg (0.666 lb)		
Wire size	0.14…2.5 mm^2 (26…14 AWG) solid copper wire or 0.14…1.5 mm^2 (26…14 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max		
Wiring category(1)	2 – on signal ports 2 – on power ports		
Wire type	Use Copper Conductors only		
Terminal screw torque	0.6 Nm (4.4 lb-in.) max (using a 2.5 mm (0.10 in.) flat-blade screwdriver)		
Input circuit type	120V AC	12/24V sink/source (standard) 24V sink/source (high-speed)	
Output circuit type	Relay		12/24V DC sink transistor (standard and high-speed)
Event input interrupt support	Yes		
Power consumption	7.88 W		
Power supply voltage range	20.4…26.4V DC Class 2		
I/O rating	Input 120V AC, 16 mA Output 2 A, 240V AC, general use	Input 24V DC, 8.8 mA Output 2 A, 240V AC, general use	Input 24V DC, 8.8 mA Output 24V DC, 1 A per point (Surrounding air temperature 30 °C) 24V DC, 0.3 A per point (Surrounding air temperature 65 °C)
Isolation voltage	250V (continuous), Reinforced Insulation Type, Outputs to Aux and Network, Inputs to Outputs 2080-LC30-16AWB: Type tested for 60 s @ 3250V DC I/O to Aux and Network, Inputs to Outputs 2080-LC30-16QWB: Type tested for 60 s @ 720V DC, Inputs to Aux and Network, 3250V DC Outputs to Aux and Network, Inputs to Outputs		50V (continuous), Reinforced Insulation Type, I/O to Aux and Network, Inputs to Outputs Type tested for 60s @ 720 V DC, I/O to Aux and Network, Inputs to Outputs
Pilot duty rating	C300, R150		–
Insulation stripping length	7 mm (0.28 in.)		
Enclosure type rating	Meets IP20		
North American temp code	T4		

(1) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

Inputs

Attribute	120V AC Input (2080-LC30-16AWB only)	High-Speed DC Input (2080-LC30-16QVB and 2080-LC30-16QWB only) (Inputs 0…3)	Standard DC Input (2080-LC30-16QVB and 2080-LC30-16QWB only) (Inputs 4…9)
Number of Inputs	10	4	6
Input group to backplane isolation	Verified by the following dielectric tests: 1,400V AC for 2 s 132V working voltage (IEC Class 2 reinforced insulation)	Verified by the following dielectric tests: 1,414V DC for 2 s 75V DC working voltage (IEC Class 2 reinforced insulation)	
Voltage category	110V AC	24V DC sink/source	
On-state voltage range	79…132V AC 47…63 Hz	16.8…26.4V DC	10…26.4V DC
Off-state voltage, max	20V AC	5V DC	
Off-state current, max	1.5 mA		
On-state current, min	5 mA @ 79V AC	5.0 mA @ 16.8V DC	1.8 mA @ 10V DC
On-state current, nom	12 mA @ 120V AC	7.66 mA @ 24V	6.15 mA @ 24V
On-state current, max	16 mA @ 132V AC	12.0 mA @ 30V DC	
Nominal impedance	12 kΩ @ 50 Hz 10 kΩ @ 60 Hz	3 kΩ	3.74 kΩ
Inrush current, max	250 mA @ 120V AC	—	
Turn on time/ Turn off time, max (without filtering)	ON: 1 ms OFF: 8 ms	ON: 3.2 µs OFF: 0.6 µs	ON: 33 µs…0.1 ms OFF: 22 µs…0.02 ms
IEC input compatibility	Type 3		
AC input filter setting	8 ms for all embedded inputs (In Connected Components Workbench, go to the Embedded I/O configuration window to reconfigure the filter setting for each input group)		

Isolated AC Inputs (2080-LC30-16QWB, 2080-LC30-16QVB) (Inputs 0…3)

Attribute	Value
On-state voltage, nom	12/24V AC @ 50/60 Hz
Off-state voltage, min	4V AC @ 50/60 Hz
Operating frequency, nom	50/60 Hz

Outputs

Attribute	Relay Output (2080-LC30-16AWB, 2080-LC30-16QWB only)	Hi-Speed Output (2080-LC30-16QVB only) (Outputs 0…1)	Standard Output (2080-LC30-16QVB only) (Outputs 2…5)
Number of outputs	6	2	4
Output voltage, min	5V DC, 5V AC	10.8V DC	10V DC
Output voltage, max	125V DC, 265V AC	26.4V DC	26.4V DC
Load current, min	10 mA	10 mA	10 mA

Outputs

Attribute	Relay Output (2080-LC30-16AWB, 2080-LC30-16QWB only)	Hi-Speed Output (2080-LC30-16QVB only) (Outputs 0…1)	Standard Output (2080-LC30-16QVB only) (Outputs 2…5)
Load current, max	2.0 A	100 mA (high-speed operation) 1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)	1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)
Surge current, per point	Refer to Relay Contacts Ratings on page 173	4.0 A every 1 s @ 30 °C; every 2 s @ 65 °C(1)	
Current, per common, max	5 A	—	—
Turn on time/ Turn off time, max	10 ms	2.5 µs	ON: 0.1 ms OFF: 1 ms

(1) Applies for general purpose operation only. Does not apply for high-speed operation.

Relay Contacts Ratings

Maximum Volts	Amperes		Amperes Continuous	Volt-Amperes	
	Make	Break		Make	Break
120V AC	15 A	1.5 A	2.0 A	1800V A	180V A
240V AC	7.5 A	0.75 A			
24V DC	1.0 A		1.0 A	28V A	
125V DC	0.22 A				

Environmental Specifications

Attribute	Value
Temperature, operating	IEC 60068-2-1 (Test Ad, Operating Cold), IEC 60068-2-2 (Test Bd, Operating Dry Heat), IEC 60068-2-14 (Test Nb, Operating Thermal Shock): -20…65 °C (-4…149 °F)
Temperature, surrounding air, max	65 °C (149 °F)
Temperature, non-operating	IEC 60068-2-1 (Test Ab, Unpackaged Nonoperating Cold), IEC 60068-2-2 (Test Bb, Unpackaged Nonoperating Dry Heat), IEC 60068-2-14 (Test Na, Unpackaged Nonoperating Thermal Shock): -40…85 °C (-40…185 °F)
Relative humidity	IEC 60068-2-30 (Test Db, Unpackaged Damp Heat): 5…95% non-condensing
Vibration	IEC 60068-2-6 (Test Fc, Operating): 2 g @ 10…500 Hz
Shock, operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): 25 g
Shock, nonoperating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): DIN mount: 25 g PANEL mount: 45 g
Emissions	CISPR 11 Group 1, Class A

Environmental Specifications

Attribute	Value
ESD immunity	IEC 61000-4-2: 6 kV contact discharges 8 kV air discharges
Radiated RF immunity	IEC 61000-4-3: 10V/m with 1 kHz sine-wave 80% AM from 80…2000 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 900 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 1890 MHz 10V/m with 1 kHz sine-wave 80% AM from 2000…2700 MHz
EFT/B immunity	IEC 61000-4-4: ±2 kV @ 5 kHz on power ports ±2 kV @ 5 kHz on signal ports
Surge transient immunity	IEC 61000-4-5: ±1 kV line-line(DM) and ±2 kV line-earth(CM) on power ports ±1 kV line-line(DM) and ±2 kV line-earth(CM) on signal ports
Conducted RF immunity	IEC 61000-4-6: 10V rms with 1 kHz sine-wave 80% AM from 150 kHz…80 MHz

Certifications

Certification (when product is marked)(1)	Value
c-UL-us	UL Listed Industrial Control Equipment, certified for US and Canada. See UL File E322657. UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada. See UL File E334470.
CE	European Union 2004/108/EC EMC Directive, compliant with: EN 61326-1; Meas./Control/Lab., Industrial Requirements EN 61000-6-2; Industrial Immunity EN 61000-6-4; Industrial Emissions EN 61131-2; Programmable Controllers (Clause 8, Zone A & B) European Union 2006/95/EC LVD, compliant with: EN 61131-2; Programmable Controllers (Clause 11)
C-Tick	Australian Radiocommunications Act, compliant with: AS/NZS CISPR 11; Industrial Emissions

(1) See the Product Certification link at http://www.rockwellautomation.com/products/certification/ for Declaration of Conformity, Certificates, and other certification details.

Micro830 24-Point Controllers

General Specifications – 2080-LC30-24QWB, 2080-LC30-24QVB, 2080-LC30-24QBB

Attribute	2080-LC30-24QWB	2080-LC30-24QVB	2080-LC30-24QBB
Number of I/O	24 (14 inputs, 10 outputs)		
Dimensions HxWxD	90 x 150 x 80 mm (3.54 x 5.91 x 3.15 in.)		
Shipping weight, approx.	0.423 kg (0.933 lb)		

General Specifications – 2080-LC30-24QWB, 2080-LC30-24QVB, 2080-LC30-24QBB

Attribute	2080-LC30-24QWB	2080-LC30-24QVB	2080-LC30-24QBB
Wire size	0.2…2.5 mm^2 (24…12 AWG) solid copper wire or 0.2…2.5 mm^2 (24…12 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max		
Wiring category(1)	2 – on signal ports 2 – on power ports		
Wire type	Use Copper Conductors only		
Terminal screw torque	0.6 Nm (4.4 lb-in) max (using a 2.5 mm (0.10 in.) flat-blade screwdriver)		
Input circuit type	12/24V sink/source (standard) 24V sink/source (high-speed)		
Output circuit type	Relay	24V DC sink (standard and high-speed)	24V DC source (standard and high-speed)
Event input interrupt support	Yes		
Power consumption	12.32 W		
Power supply voltage range	20.4…26.4V DC Class 2		
I/O rating	Input 24V DC, 8.8 mA Output 2 A, 240V AC, general use	Input 24V DC, 8.8 mA Output 24V DC, Class 2, 1 A per point (Surrounding air temperature 30 °C) 24 V DC, Class 2, 0.3 A per point (Surrounding air temperature 65 °C)	
Isolation voltage	250V (continuous), Reinforced Insulation Type, Outputs to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720V DC, Inputs to Aux and Network, 3250 V DC Outputs to Aux and Network, Inputs to Outputs	50V (continuous), Reinforced Insulation Type, I/O to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720V DC, I/O to Aux and Network, Inputs to Outputs	
Pilot duty rating	C300, R150 (2080-LC30-24QWB only)	—	
Insulation stripping length	7 mm (0.28 in.)		
Enclosure type rating	Meets IP20		
North American temp code	T4		

(1) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

Inputs

Attribute	High-Speed DC Input (Inputs 0…7)	Standard DC Input (Inputs 8 and higher)
Number of Inputs	8	6
Voltage category	24V DC sink/source	
Operating voltage range	16.8…26.4V DC	10…26.4V DC
Off-state voltage, max	5V DC	
Off-state current, max	1.5 mA	
On-state current, min	5.0 mA @ 16.8V DC	1.8 mA @ 10V DC
On-state current, nom	8.8 mA @ 24V DC	8.5 mA @ 24V DC
On-state current, max	12.0 mA @ 30V DC	

Inputs

Attribute	High-Speed DC Input (Inputs 0…7)	Standard DC Input (Inputs 8 and higher)
Nominal impedance	3 kΩ	3.74 kΩ
IEC input compatibility	Type 3	
AC input filter setting	8 ms for all embedded inputs (In Connected Components Workbench, go to the Embedded I/O configuration window to re-configure the filter setting for each input group)	

Isolated AC Inputs (2080-LC30-24QWB, 2080-LC30-24QVB, 2080-LC30-24QBB) (Inputs 0…7)

Attribute	Value
On-state voltage, nom	12/24V AC @ 50/60 Hz
Off-state voltage, min	4V AC @ 50/60Hz
Operating frequency, nom	50/60 Hz

Outputs

Attribute	2080-LC30-24QWB	2080-LC30-24QVB / 2080-LC30-24QBB	
	Relay Output	Hi-Speed Output (Outputs 0…1)	Standard Output (Outputs 2 and higher)
Number of outputs	10	2	8
Output voltage, min	5V DC, 5V AC	10.8V DC	10V DC
Output voltage, max	125V DC, 265V AC	26.4V DC	26.4V DC
Load current, min	10 mA		
Load current, max	2.0 A	100 mA (high-speed operation) 1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)	1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)
Surge current, per point	Refer to Relay Contacts Ratings on page 176	4.0 A every 1 s @ 30 °C; every 2 s @ 65 °C(1)	
Current, per common, max	5 A	—	—
Turn on time/ Turn off time, max	10 ms	2.5 µs	0.1 ms 1 ms

(1) Applies for general purpose operation only. Does not apply for high-speed operation.

Relay Contacts Ratings

Maximum Volts	Amperes		Amperes Continuous	Volt-Amperes	
	Make	Break		Make	Break
120V AC	15 A	1.5 A	2.0 A	1800V A	180V A
240V AC	7.5 A	0.75 A			
24V DC	1.0 A		1.0 A	28V A	
125V DC	0.22 A				

Environmental Specifications

Attribute	Value
Temperature, operating	IEC 60068-2-1 (Test Ad, Operating Cold), IEC 60068-2-2 (Test Bd, Operating Dry Heat), IEC 60068-2-14 (Test Nb, Operating Thermal Shock): -20…65 °C (-4…149 °F)
Temperature, surrounding air, max	65 °C (149 °F)
Temperature, non-operating	IEC 60068-2-1 (Test Ab, Unpackaged Nonoperating Cold), IEC 60068-2-2 (Test Bb, Unpackaged Nonoperating Dry Heat), IEC 60068-2-14 (Test Na, Unpackaged Nonoperating Thermal Shock): -40…85 °C (-40…185 °F)
Relative humidity	IEC 60068-2-30 (Test Db, Unpackaged Damp Heat): 5…95% non-condensing
Vibration	IEC 60068-2-6 (Test Fc, Operating): 2 g @ 10…500 Hz
Shock, operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): 25 g
Shock, non-operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): DIN mount: 25 g PANEL mount: 35 g
Emissions	CISPR 11 Group 1, Class A
ESD immunity	IEC 61000-4-2: 6 kV contact discharges 8 kV air discharges
Radiated RF immunity	IEC 61000-4-3: 10V/m with 1 kHz sine-wave 80% AM from 80…2000 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 900 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 1890 MHz 10V/m with 1 kHz sine-wave 80% AM from 2000…2700 MHz
EFT/B immunity	IEC 61000-4-4: ±2 kV at 5 kHz on power ports ±2 kV at 5 kHz on signal ports
Surge transient immunity	IEC 61000-4-5: ±1 kV line-line(DM) and ±2 kV line-earth(CM) on power ports ±1 kV line-line(DM) and ±2 kV line-earth(CM) on signal ports
Conducted RF immunity	IEC 61000-4-6: 10V rms with 1 kHz sine-wave 80% AM from 150 kHz…80 MHz

Certifications

Certification (when product is marked)(1)	Value
c-UL-us	UL Listed Industrial Control Equipment, certified for US and Canada. See UL File E322657. UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada. See UL File E334470.
CE	European Union 2004/108/EC EMC Directive, compliant with: EN 61326-1; Meas./Control/Lab., Industrial Requirements EN 61000-6-2; Industrial Immunity EN 61000-6-4; Industrial Emissions EN 61131-2; Programmable Controllers (Clause 8, Zone A & B) European Union 2006/95/EC LVD, compliant with: EN 61131-2; Programmable Controllers (Clause 11)
C-Tick	Australian Radiocommunications Act, compliant with: AS/NZS CISPR 11; Industrial Emissions

(1) See the Product Certification link at http://www.rockwellautomation.com/products/certification/ for Declaration of Conformity, Certificates, and other certification details.

Micro830 48-Point Controllers

General Specifications – 2080-LC30-48AWB, 2080-LC30-48QWB, 2080-LC30-48QVB, 2080-LC30-48QBB

Attribute	2080-LC30-48AWB	2080-LC30-48QWB	2080-LC30-48QVB	2080-LC30-48QBB
Number of I/O	48 (28 inputs, 20 outputs)			
Dimensions HxWxD	90 x 230 x 80 mm (3.54 x 9.06 x 3.15 in.)			
Shipping weight, approx.	0.725 kg (1.60 lb)			
Wire size	0.2…2.5 mm^2 (24…12 AWG) solid copper wire or 0.2…2.5 mm^2 (24…12 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max			
Wiring category(1)	2 – on signal ports 2 – on power ports			
Wire type	Use copper conductors only			
Terminal screw torque	0.6 Nm (4.4 lb-in) max (using a 2.5 mm (0.10 in.) flat-blade screwdriver)			
Input circuit type	120V AC	12/24V sink/source (standard) 24V sink/source (high-speed)		
Output circuit type	Relay		24V DC sink (standard and high-speed)	24V DC source (standard and high-speed)
Event input interrupt support	Yes, inputs 0…15 only			
Power consumption	18.2 W			
Power supply voltage range	20.4…26.4V DC Class 2			

General Specifications – 2080-LC30-48AWB, 2080-LC30-48QWB, 2080-LC30-48QVB, 2080-LC30-48QBB

Attribute	2080-LC30-48AWB	2080-LC30-48QWB	2080-LC30-48QVB	2080-LC30-48QBB
I/O rating	Input 120V AC, 16 mA Output 2 A, 240V AC, general use	Input 24V DC, 8.8 mA Output 2 A, 240V AC, general use	Input 24V DC, 8.8 mA Output 24V DC, 1 A per point (Surrounding air temperature 30 °C) 24 V DC, 0.3 A per point (Surrounding air temperature 65 °C)	
Insulation stripping length	7 mm (0.28 in.)			
Enclosure type rating	Meets IP20			
Pilot duty rating	C300, R150		—	
Isolation voltage	250V (continuous), Reinforced Insulation Type, Outputs to Aux and Network, Inputs to Outputs Type tested for 60 s @ 3250V DC I/O to Aux and Network, Inputs to Outputs	250V (continuous), Reinforced Insulation Type, Outputs to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720V DC, Inputs to Aux and Network, 3250V DC Outputs to Aux and Network, Inputs to Outputs	50V (continuous), Reinforced Insulation Type, I/O to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720V DC, I/O to Aux and Network, Inputs to Outputs	
North American temp code	T4			

(1) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

Inputs

Attribute	2080-LC30-48AWB	2080-LC30-48QWB / 2080-LC30-48QVB / 2080-LC30-48QBB	
	120V AC Input	High-Speed DC Input (Inputs 0…11)	Standard DC Input (Inputs 12 and higher)
Number of Inputs	28	12	16
Voltage category	110V AC	24V DC sink/source	
Operating voltage	132V, 60Hz AC, max	16.8…26.4V DC	10…26.4V DC
Off-state voltage, max	20V AC	5V DC	
Off-state current, max	1.5 mA	1.5 mA	
On-state current, min	5 mA @ 79V AC	5.0 mA @ 16.8V DC	1.8 mA @ 10V DC
On-state current, nom	12 mA @ 120V AC	8.8 mA @ 24V DC	8.5 mA @ 24V DC
On-state current, max	16 mA @ 132V AC	12.0 mA @ 30V DC	
Nominal impedance	12 kΩ @ 50 Hz 10 kΩ @ 60 Hz	3 kΩ	3.74 kΩ
IEC input compatibility	Type 3		
Inrush current, max	250 mA @ 120V AC		
Input frequency, max	63 Hz		
AC input filter setting	8 ms for all embedded inputs (In Connected Components Workbench, go to the Embedded I/O configuration window to reconfigure the filter setting for each input group)		

Isolated AC Inputs (2080-LC30-48QWB, 2080-LC30-48QVB, 2080-LC30-48QBB) (Inputs 0…11)

Attribute	Value
On-state voltage, nom	12/24V AC @ 50/60 Hz
Off-state voltage, min	4V AC @ 50/60Hz
Operating frequency, nom	50/60 Hz

Outputs

Attribute	2080-LC30-48AWB / 2080-L30-48QWB	2080-LC30-48QVB / 2080-LC30-48QBB	
	Relay Output	Hi-Speed Output (Outputs 0…3)	Standard Output (Outputs 4 and higher)
Number of outputs	20	4	16
Output voltage, min	5V DC, 5V AC	10.8V DC	10V DC
Output voltage, max	125V DC, 265V AC	26.4V DC	26.4V DC
Load current, min	10 mA		
Load current, max	2.0 A	100 mA (high-speed operation) 1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)	1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)
Surge current, per point	Refer to Relay Contacts Ratings on page 180	4.0 A every 1 s @ 30 °C; every 2 s @ 65 °C(1)	
Current, per common, max	5 A	—	—
Turn on time/ Turn off time, max	10 ms	2.5 µs	0.1 ms 1 ms

(1) Applies for general purpose operation only. Does not apply for high-speed operation.

Relay Contacts Ratings

Maximum Volts	Amperes		Amperes Continuous	Volt-Amperes	
	Make	Break		Make	Break
120V AC	15 A	1.5 A	2.0 A	1800V A	180V A
240V AC	7.5 A	0.75 A			
24V DC	1.0 A		1.0 A	28V A	
125V DC	0.22 A				

Environmental Specifications

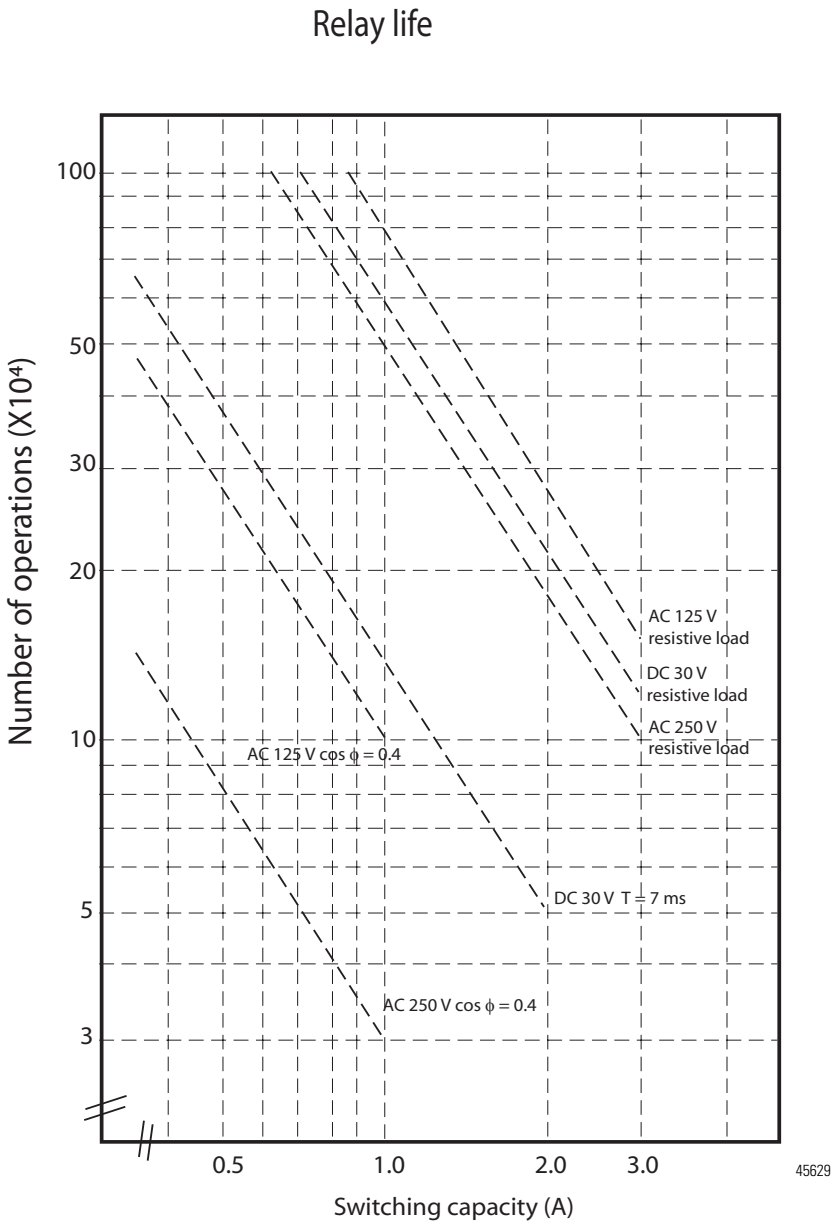
Attribute	Value
Temperature, operating	IEC 60068-2-1 (Test Ad, Operating Cold), IEC 60068-2-2 (Test Bd, Operating Dry Heat), IEC 60068-2-14 (Test Nb, Operating Thermal Shock): -20…65 °C (-4…149 °F)
Temperature, surrounding air, max	65 °C (149 °F)
Temperature, non-operating	IEC 60068-2-1 (Test Ab, Unpackaged Nonoperating Cold), IEC 60068-2-2 (Test Bb, Unpackaged Nonoperating Dry Heat), IEC 60068-2-14 (Test Na, Unpackaged Nonoperating Thermal Shock): -40…85 °C (-40…185 °F)
Relative humidity	IEC 60068-2-30 (Test Db, Unpackaged Damp Heat): 5…95% non-condensing
Vibration	IEC 60068-2-6 (Test Fc, Operating): 2 g @ 10…500 Hz
Shock, operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): 25 g
Shock, non-operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): DIN mount: 25 g PANEL mount: 35 g
Emissions	CISPR 11 Group 1, Class A
ESD immunity	IEC 61000-4-2: 6 kV contact discharges 8 kV air discharges
Radiated RF immunity	IEC 61000-4-3: 10V/m with 1 kHz sine-wave 80% AM from 80…2000 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 900 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 1890 MHz 10V/m with 1 kHz sine-wave 80% AM from 2000…2700 MHz
EFT/B immunity	IEC 61000-4-4: ±2 kV @ 5 kHz on power ports ±2 kV @ 5 kHz on signal ports
Surge transient immunity	IEC 61000-4-5: ±1 kV line-line(DM) and ±2 kV line-earth(CM) on power ports ±1 kV line-line(DM) and ±2 kV line-earth(CM) on signal ports
Conducted RF immunity	IEC 61000-4-6: 10V rms with 1 kHz sine-wave 80% AM from 150 kHz…80 MHz

Certifications

Certification (when product is marked)(1)	Value
c-UL-us	UL Listed Industrial Control Equipment, certified for US and Canada. See UL File E322657. UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada. See UL File E334470.
CE	European Union 2004/108/EC EMC Directive, compliant with: EN 61326-1; Meas./Control/Lab., Industrial Requirements EN 61000-6-2; Industrial Immunity EN 61000-6-4; Industrial Emissions EN 61131-2; Programmable Controllers (Clause 8, Zone A & B) European Union 2006/95/EC LVD, compliant with: EN 61131-2; Programmable Controllers (Clause 11)
C-Tick	Australian Radiocommunications Act, compliant with: AS/NZS CISPR 11; Industrial Emissions

(1) See the Product Certification link at http://www.rockwellautomation.com/products/certification/ for Declaration of Conformity, Certificates, and other certification details.

Micro830 and Micro850 Relay Charts



Micro850 Controllers

The following tables provide specifications, ratings, and certifications for the 24-point and 48-point Micro850 controllers.

Micro850 24-Point Controllers

General Specifications – 2080-LC50-24AWB, 2080-LC50-24QWB, 2080-LC50-24QVB, 2080-LC50-24QBB

Attribute	2080-LC50-24AWB	2080-LC50-24QWB	2080-LC50-24QVB	2080-LC50-24QBB
Number of I/O	24 (14 inputs, 10 outputs)			
Dimensions HxWxD	90 x 158 x 80 mm (3.54 x 6.22 x 3.15 in.)			
Shipping weight, approx.	0.423 kg (0.933 lb)			
Wire size	0.2…2.5 mm^2 (24…12 AWG) solid copper wire or 0.2…2.5 mm^2 (24…12 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max			
Wiring category(1)	2 – on signal ports 2 – on power ports 2 – on communication ports			
Wire type	Use Copper Conductors only			
Terminal screw torque	0.4…0.5 Nm (3.5…4.4 lb-in.) using a 0.6 x 3.5 mm flat-blade screwdriver. (Note: Use a handheld screwdriver to hold down the screws at the side.)			
Input circuit type	12/24V sink/source (standard) 24V sink/source (high-speed)			
Output circuit type	Relay		24V DC sink (standard and high-speed)	24V DC source (standard and high-speed)
Power consumption	28 W			
Power supply voltage range	20.4…26.4V DC Class 2			
I/O rating	Input 120V AC 16 mA Output 2 A, 240V AC, 2A, 24V DC	Input 24V, 8.8 mA Output 2 A, 240V AC 2A, 24V DC	Input 24V, 8.8 mA Output 24V DC, Class 2, 1 A per point (Surrounding air temperature 30 °C) 24V DC, Class 2, 0.3 A per point (Surrounding air temperature 65 °C)	
Isolation voltage	250V (continuous), Reinforced Insulation Type, Output to Aux and Network, Inputs to Outputs. Type tested for 60 s @ 3250V DC Output to Aux and Network, Inputs to Outputs 150V (continuous), Reinforced Insulation Type, Input to Aux and Network. Type tested for 60 s @ 1950V DC Input to Aux and Network	250V (continuous), Reinforced Insulation Type, Output to Aux and Network, Inputs to Outputs. Type tested for 60 s @ 3250V DC Output to Aux and Network, Inputs to Outputs. 50V (continuous), Reinforced Insulation Type, Input to Aux and Network Type tested for 60 s @ 720V DC, Input to Aux and Network	50V (continuous), Reinforced Insulation Type, I/O to Aux and Network, Inputs to Outputs. Type tested for 60 s @ 720 V DC, I/O to Aux and Network, Inputs to Outputs.	
Pilot duty rating	C300, R150		–	
Insulation stripping length	7 mm (0.28 in.)			
Enclosure type rating	Meets IP20			
North American temp code	T4			

(1) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

DC Input Specifications – 2080-LC50-24QBB, 2080-LC50-24QVB, 2080-LC50-24QWB

Attribute	High-Speed DC Input (Inputs 0…7)	Standard DC Input (Inputs 8 and higher)
Number of Inputs	8	6
Voltage category	24V sink/source	
Input group to backplane isolation	Verified by one of the following dielectric tests: 720V DC for 2 s 50V DC working voltage (IEC Class 2 reinforced insulation)	
On-state voltage range	16.8…26.4V DC @ 65°C (149°F) 16.8…30.0V DC @ 30°C (86°F)	10…26.4V DC @ 65°C (149°F) 10…30.0V DC @ 30°C (86°F)
Off-state voltage	5V DC, max	
Off-state current	1.5 mA, max	
On-state current	5.0 mA @ 16.8V DC, min 7.6 mA @ 24V DC, nom 12.0 mA @ 30V DC, max	1.8 mA @ 10V DC, min 6.15 mA @ 24V DC, nom 12.0 mA @ 30V DC, max
Nominal impedance	3 kΩ	3.74 kΩ
IEC input compatibility	Type 3	

AC Input Specifications – 2080-LC50-24AWB

Attribute	Value
Number of Inputs	14
On-state voltage	79 V AC, min 132V AC, max
On-state current	5 mA, min 16 mA, max
Input frequency	50/60 Hz, nom 47 Hz, min 63 Hz, max
Off-state voltage	20V AC @ 120V AC, max
Off-state current	2.5 mA @ 120V AC, max
Inrush current	250 mA @ 120V AC, max
Inrush delay time constant max	22 ms
IEC input compatibility	Type 3

Output Specifications

Attribute	2080-LC50-24QWB 2080-LC50-24AWB	2080-LC50-24QVB / 2080-LC50-24QBB	
	Relay Output	Hi-Speed Output (Outputs 0...1)	Standard Output (Outputs 2 and higher)
Number of outputs	10	2	8
Output voltage, min	5V DC, 5V AC	10.8V DC	10V DC
Output voltage, max	125V DC, 265V AC	26.4V DC	26.4V DC
Load current, min	10 mA		

Output Specifications

Attribute	2080-LC50-24QWB 2080-LC50-24AWB	2080-LC50-24QVB / 2080-LC50-24QBB	
	Relay Output	Hi-Speed Output (Outputs 0...1)	Standard Output (Outputs 2 and higher)
Load current, continuous, max	2.0 A	100 mA (high-speed operation) 1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)	1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)
Surge current, per point	See Relay Contacts Ratings on page 176	4.0 A for 10 ms every 1 s @ 30 °C; every 2 s @ 65 °C(1)	
Current, per common, max	5 A	–	–
Turn on time/ Turn off time, max	10 ms	2.5 μs	0.1 ms 1 ms

(1) Applies for general purpose operation only; does not apply for high-speed operation.

Relay Contacts Ratings

Maximum Volts	Amperes		Amperes Continuous	Volt-Amperes	
	Make	Break		Make	Break
120V AC	15 A	1.5 A	2.0 A	1800V A	180V A
240V AC	7.5 A	0.75 A			
24V DC	1.0 A		1.0 A	28V A	
125V DC	0.22 A				

Environmental Specifications

Attribute	Value
Temperature, operating	IEC 60068-2-1 (Test Ad, Operating Cold), IEC 60068-2-2 (Test Bd, Operating Dry Heat), IEC 60068-2-14 (Test Nb, Operating Thermal Shock): -20…65 °C (-4…149 °F)
Temperature, surrounding air, max	65 °C (149 °F)
Temperature, non-operating	IEC 60068-2-1 (Test Ab, Unpackaged Nonoperating Cold), IEC 60068-2-2 (Test Bb, Unpackaged Nonoperating Dry Heat), IEC 60068-2-14 (Test Na, Unpackaged Nonoperating Thermal Shock): -40…85 °C (-40…185 °F)
Relative humidity	IEC 60068-2-30 (Test Db, Unpackaged Damp Heat): 5…95% non-condensing
Vibration	IEC 60068-2-6 (Test Fc, Operating): 2 g @ 10…500 Hz
Shock, operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): 25 g
Shock, non-operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): DIN mount: 25 g PANEL mount: 35 g
Emissions	CISPR 11 Group 1, Class A

Environmental Specifications

Attribute	Value
ESD immunity	IEC 61000-4-2: 6 kV contact discharges 8 kV air discharges
Radiated RF immunity	IEC 61000-4-3: 10V/m with 1 kHz sine-wave 80% AM from 80…2000 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 900 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 1890 MHz 10V/m with 1 kHz sine-wave 80% AM from 2000…2700 MHz
EFT/B immunity	IEC 61000-4-4: ±2 kV @ 5 kHz on power ports ±2 kV @ 5 kHz on signal ports ±1 kV @ 5 kHz on communication ports
Surge transient immunity	IEC 61000-4-5: ±1 kV line-line(DM) and ±2 kV line-earth(CM) on power ports ±1 kV line-line(DM) and ±2 kV line-earth(CM) on signal ports ±1 kV line-earth(CM) on communication ports
Conducted RF immunity	IEC 61000-4-6: 10V rms with 1 kHz sine-wave 80% AM from 150 kHz…80 MHz

Isolated AC Inputs (2080-LC50-24QWB, 2080-LC50-24QVB, 2080-LC50-24QBB) (Inputs 0…7)

Attribute	Value
On-state voltage, nom	12/24V AC @ 50/60 Hz
Off-state voltage, min	4V AC @ 50/60Hz
Operating frequency, nom	50/60 Hz

Micro850 48-Point Controllers

General Specifications – 2080-LC50-48AWB, 2080-LC50-48QWB, 2080-LC50-48QVB, 2080-LC50-48QBB

Attribute	2080-LC50-48AWB	2080-LC50-48QWB	2080-LC50-48QVB	2080-LC50-48QBB
Number of I/O	48 (28 inputs, 20 outputs)			
Dimensions HxWxD	90 x 238 x 80 mm (3.54 x 9.37 x 3.15 in.)			
Shipping weight, approx.	0.725 kg (1.60 lb)			
Wire size	0.2…2.5 mm^2 (24…12 AWG) solid copper wire or 0.2…2.5 mm^2 (24…12 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max			
Wiring category(1)	2 – on signal ports 2 – on power ports 2 – on communication ports			
Wire type	Use Copper Conductors only			
Terminal screw torque	0.4…0.5 Nm (3.5…4.4 lb-in.) (using a 0.6 x 3.5 mm flat-blade screwdriver)			
Input circuit type	120V AC	12/24V sink/source (standard) 24V sink/source (high-speed)		
Output circuit type	Relay		24V DC sink (standard and high-speed)	24V DC source (standard and high-speed)

General Specifications – 2080-LC50-48AWB, 2080-LC50-48QWB, 2080-LC50-48QVB, 2080-LC50-48QBB

Attribute	2080-LC50-48AWB	2080-LC50-48QWB	2080-LC50-48QVB	2080-LC50-48QBB
Power consumption	33 W			
Power supply voltage range	20.4…26.4V DC Class 2			
I/O rating	Input 120V AC, 16 mA Output 2 A, 240V AC, 2 A, 24V DC	Input 24V, 8.8 mA Output 2 A, 240V AC, 2 A, 24V DC	Input 24V, 8.8 mA Output 24V DC, 1 A per point (surrounding air temperature 30 °C) 24V DC, 0.3 A per point (surrounding air temperature 65 °C)	
Insulation stripping length	7 mm (0.28 in)			
Enclosure type rating	Meets IP20			
Pilot duty rating	C300, R150		–	
Isolation voltage	250V (continuous), Reinforced Insulation Type, Output to Aux and Network, Inputs to Outputs. Type tested for 60 s @ 3250V DC Output to Aux and Network, Inputs to Outputs. 150V (continuous), Reinforced Insulation Type, Input to Aux and Network Type tested for 60 s @ 1950V DC Input to Aux and Network.	250V (continuous), Reinforced Insulation Type, Output to Aux and Network, Inputs to Outputs Type tested for 60 s @ 3250V DC Output to Aux and Network, Inputs to Outputs 50V (continuous), Reinforced Insulation Type, Input to Aux and Network Type tested for 60 s @ 720V DC, Inputs to Aux and Network	50V (continuous), Reinforced Insulation Type, I/O to Aux and Network, Inputs to Outputs Type tested for 60 s @ 720V DC, I/O to Aux and Network, Inputs to Outputs.	
North American temp code	T4			

(1) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

Input Specifications

Attribute	2080-LC50-48AWB	2080-LC50-48QWB / 2080-LC50-48QVB / 2080-LC50-48QBB	
	120V AC Input	High-Speed DC Input (Inputs 0…11)	Standard DC Input (Inputs 12 and higher)
Number of Inputs	28	12	16
Input group to backplane isolation	Verified by the following dielectric tests: 1950V AC for 2 s 150V working voltage (IEC Class 2 reinforced insulation)	Verified by the following dielectric tests: 720V DC for 2 s 50V DC working voltage (IEC Class 2 reinforced insulation)	
Voltage category	110V AC	24V DC sink/source	
Operating voltage range	132V, 60Hz AC max	16.8…26.4V DC @ 65°C (149°F) 16.8…30.0V DC @ 30°C (86°F)	10…26.4V DC @ 65°C (149°F) 10…30.0V DC @ 30°C (86°F)
Off-state voltage, max	20V AC	5V DC	
Off-state current, max	1.5 mA	1.5 mA	
On-state current, min	5 mA @ 79V AC	5.0 mA @ 16.8V DC	1.8 mA @ 10V DC
On-state current, nom	12 mA @ 120V AC	7.6 mA @ 24V DC	6.15 mA @ 24V DC
On-state current, max	16 mA @ 132V AC	12.0 mA @ 30V DC	
Nominal impedance	12 kΩ @ 50 Hz 10 kΩ @ 60 Hz	3 kΩ	3.74 kΩ
IEC input compatibility	Type 3		
Inrush current, max	250 mA @ 120V AC	–	
Input frequency, max	63 Hz	–	

Output Specifications

Attribute	2080-LC50-48AWB / 2080-LC50-48QWB	2080-LC50-48QVB / 2080-LC50-48QBB	
	Relay Output	Hi-Speed Output (Outputs 0…3)	Standard Output (Outputs 4 and higher)
Number of outputs	20	4	16
Output voltage, min	5V DC, 5V AC	10.8V DC	10V DC
Output voltage, max	125V DC, 265V AC	26.4V DC	26.4V DC
Load current, min	10 mA		
Load current, continuous, max	2.0 A	100 mA (high-speed operation) 1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)	1.0 A @ 30 °C 0.3 A @ 65 °C (standard operation)
Surge current, per point	See Relay Contacts Ratings on page 180	4.0 A for 10 ms every 1 s @ 30 °C; every 2 s @ 65 °C(1)	
Current, per common, max	5A	–	–
Turn on time/ Turn off time, max	10 ms	2.5 µs	0.1 ms 1 ms

(1) Applies for general purpose operation only. Does not apply for high-speed operation

Isolated AC Inputs (2080-LC50-48QWB, 2080-LC50-48QVB, 2080-LC50-48QBB) (Inputs 0…11)

Attribute	Value
On-state voltage, nom	12/24V AC @ 50/60 Hz
Off-state voltage, min	4V AC @ 50/60Hz
Operating frequency, nom	50/60 Hz

Relay Contacts Ratings

Maximum Volts	Amperes		Amperes Continuous	Volt-Amperes	
	Make	Break		Make	Break
120V AC	15 A	1.5 A	2.0 A	1800V A	180V A
240V AC	7.5 A	0.75 A			
24V DC	1.0 A		1.0 A	28V A	
125V DC	0.22 A				

Environmental Specifications

Attribute	Value
Temperature, operating	IEC 60068-2-1 (Test Ad, Operating Cold), IEC 60068-2-2 (Test Bd, Operating Dry Heat), IEC 60068-2-14 (Test Nb, Operating Thermal Shock): -20…65 °C (-4…149 °F)
Temperature, surrounding air, max	65 °C (149 °F)
Temperature, non-operating	IEC 60068-2-1 (Test Ab, Unpackaged Nonoperating Cold), IEC 60068-2-2 (Test Bb, Unpackaged Nonoperating Dry Heat), IEC 60068-2-14 (Test Na, Unpackaged Nonoperating Thermal Shock): -40…85 °C (-40…185 °F)
Relative humidity	IEC 60068-2-30 (Test Db, Unpackaged Damp Heat): 5…95% non-condensing
Vibration	IEC 60068-2-6 (Test Fc, Operating): 2 g @ 10…500 Hz
Shock, operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): 25 g
Shock, non-operating	IEC 60068-2-27 (Test Ea, Unpackaged Shock): DIN mount: 25 g PANEL mount: 35 g
Emissions	CISPR 11 Group 1, Class A
ESD immunity	IEC 61000-4-2: 4 kV contact discharges 8 kV air discharges
Radiated RF immunity	IEC 61000-4-3: 10V/m with 1 kHz sine-wave 80% AM from 80…2000 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 900 MHz 10V/m with 200 Hz 50% Pulse 100% AM @ 1890 MHz 10V/m with 1 kHz sine-wave 80% AM from 2000…2700 MHz
EFT/B immunity	IEC 61000-4-4: ±2 kV @ 5 kHz on power ports ±2 kV @ 5 kHz on signal ports ±1 kV @ 5 kHz on communication ports
Surge transient immunity	IEC 61000-4-5: ±1 kV line-line(DM) and ±2 kV line-earth(CM) on power ports ±1 kV line-line(DM) and ±2 kV line-earth(CM) on signal ports ±1 kV line-earth(CM) on communication ports
Conducted RF immunity	IEC 61000-4-6: 10V rms with 1 kHz sine-wave 80% AM from 150 kHz…80 MHz

Certifications

Certification (when product is marked)(1)	Value
c-UL-us	UL Listed Industrial Control Equipment, certified for US and Canada. See UL File E322657. UL Listed for Class I, Division 2 Group A,B,C,D Hazardous Locations, certified for U.S. and Canada. See UL File E334470.
CE	European Union 2004/108/EC EMC Directive, compliant with: EN 61326-1; Meas./Control/Lab., Industrial Requirements EN 61000-6-2; Industrial Immunity EN 61000-6-4; Industrial Emissions EN 61131-2; Programmable Controllers (Clause 8, Zone A & B) European Union 2006/95/EC LVD, compliant with: EN 61131-2; Programmable Controllers (Clause 11)
C-Tick	Australian Radiocommunications Act, compliant with: AS/NZS CISPR 11; Industrial Emissions
EtherNet/IP	ODVA conformance tested to EtherNet/IP specifications.
KC	Korean Registration of Broadcasting and Communications Equipment, compliant with: Article 58-2 of Radio Waves Act, Clause 3.

(1) See the Product Certification link at http://www.rockwellautomation.com/products/certification for Declaration of Conformity, Certificates, and other certification details.

For the Micro850 relay chart, see Micro830 and Micro850 Relay Charts on page 183.

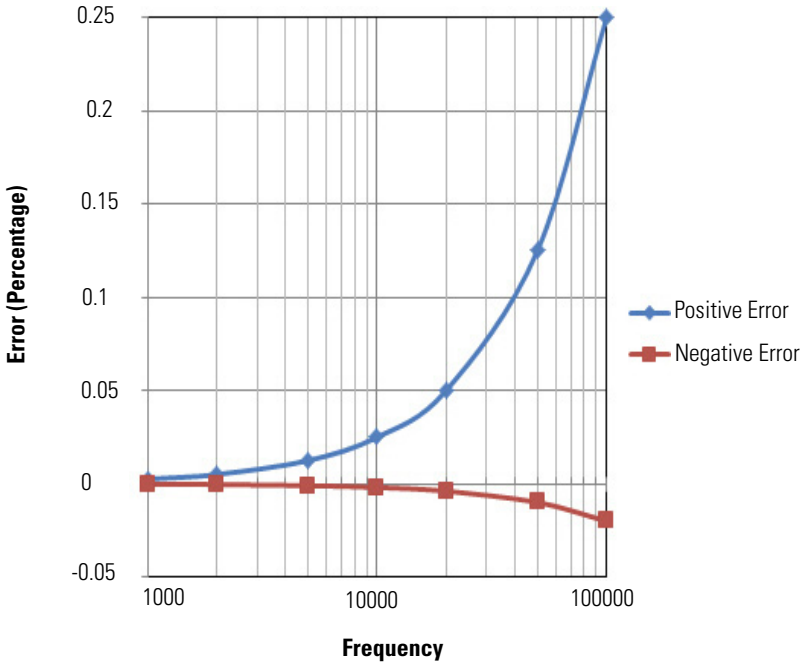
PTO Output Duty Cycle Error

Turn On/Off time for the Micro830 and Micro850 controllers for the PTO output port is 0.2 µs and 2.5 µs max, respectively. Duty cycle error is:
Positive error = 2.5 µs * F
Negative error = -0.2 µs * F

The plot below shows duty cycle error vs. frequency.

To get the duty cycle error at a certain frequency, for example, the user sets frequency to 20 kHz, and sets duty cycle to 30% in Connected Components Workbench, then actual duty cycle is

$30\%\ ^{+5\%}_{-0.4\%}$.



PTO Typical Readings

PTO Typical Readings

		Expected Duty Cycle		Typical Duty Cycle (1.27 KΩ load)
Frequency (kHz)	**%Duty Cycle**	**Minimum**	**Maximum**	**%Duty Cycle**
5	5%	4.90%	6.25%	5.48
5	10%	9.90%	11.25%	10.5
5	20%	19.90%	21.25%	20.5
5	40%	39.90%	41.25%	40.5
5	55%	54.90%	56.25%	55.5
5	65%	64.90%	66.25%	65.5
5	75%	74.90%	76.25%	75.5
5	95%	94.90%	96.25%	95.5
10	5%	4.80%	7.50%	5.9
10	10%	9.80%	12.50%	11.0
10	20%	19.80%	22.50%	21.0
10	40%	39.80%	42.50%	40.9
10	55%	54.80%	57.50%	55.9
10	65%	64.80%	67.50%	65.9
10	85%	84.80%	87.50%	85.9
10	95%	94.90%	97.50%	95.9
25	5%	4.50%	11.25%	7.25

PTO Typical Readings

		Expected Duty Cycle		Typical Duty Cycle (1.27 KΩ load)
Frequency (kHz)	**%Duty Cycle**	**Minimum**	**Maximum**	**%Duty Cycle**
25	10%	9.50%	16.25%	12.3
25	20%	19.50%	26.25%	22.4
25	40%	39.50%	46.25%	42.3
25	55%	54.50%	61.25%	57.3
25	65%	64.50%	71.25%	67.3
25	85%	84.50%	91.25%	87.3
25	95%	94.50%	100.00%	97.0
50	5%	4.00%	17.50%	9.7
50	10%	9.00%	22.50%	14.8
50	20%	19.00%	32.50%	24.7
50	40%	39.00%	52.50%	44.7
50	55%	54.00%	67.50%	59.6
50	65%	64.00%	77.50%	69.6
50	85%	84.00%	97.50%	89.5
50	95%	94.00%	100.00%	98.1
100	5%	3.00%	30.00%	14.7
100	10%	8.00%	35.00%	19.5
100	20%	18.00%	45.00%	19.6
100	40%	38.00%	65.00%	49.3
100	55%	53.00%	80.00%	64.0
100	65%	63.00%	90.00%	73.8
100	85%	83.00%	100.00%	92.4
100	95%	93.00%	100.00%	98.0

Micro800 Programmable Controller External AC Power Supply

General Specifications

Attribute	Value
Dimensions, HxWxD	90 x 45 x 80 mm (3.55 x 1.78 x 3.15 in.)
Shipping weight	0.34 kg (0.75 lb)
Supply voltage range[1]	100V…120V AC, 1 A 200…240V AC, 0.5 A
Supply frequency	47…63 Hz
Supply power	24V DC, 1.6 A
Inrush current, max	24A @ 132V for 10 ms 40A @ 263V for 10 ms

General Specifications

Attribute	Value
Power consumption (Output power)	38.4W @ 100V AC, 38.4W @ 240V AC
Power dissipation (Input power)	45.1W @ 100V AC, 44.0W @ 240V AC
Isolation voltage	250V (continuous), Primary to Secondary: Reinforced Insulation Type Type tested for 60s @ 2300V AC primary to secondary and 1480V AC primary to earth ground.
Output ratings, max	24V DC, 1.6A, 38.4W
Enclosure type rating	Meets IP20
Wire size	0.32... 2.1 mm² (22...14 AWG) solid copper wire or 0.32... 1.3 mm² (22...16 AWG) stranded copper wire rated @ 90 °C (194 °F) insulation max
Terminal screw torque	0.5…0.6 Nm (4.4…5.3 lb-in.) (using a Phillips-head or 2.5 mm (0.10in.) flat-blade screwdriver)
Wiring category(2)	2 – on power ports
Insulation stripping length	7 mm (0.28 in.)
North American temp code	T4A

(1) Any fluctuation in voltage source must be within 85V...264V. Do not connect the adapter to a power source that has fluctuations outside of this range.

(2) Use this Conductor Category information for planning conductor routing. Refer to Industrial Automation Wiring and Grounding Guidelines, publication 1770-4.1.

Modbus Mapping for Micro800

Modbus Mapping

All Micro800 controllers (except the Micro810 12-point models) support Modbus RTU over a serial port through the embedded, non-isolated serial port. The 2080-SERIALISOL isolated serial port plug-in module also supports Modbus RTU. Both Modbus RTU master and slave are supported. Although performance may be affected by the program scan time, the 48-point controllers can support up to six serial ports (one embedded and five plug-ins), and so consequently, six separate Modbus networks.

In addition, the Micro850 controller supports Modbus TCP Client/Server through the Ethernet port.

Endian Configuration

Modbus protocol is big-endian in that the most significant byte of a 16-bit word is transmitted first. Micro800 is also big-endian, so byte ordering does not have to be reversed. For Micro800 data types larger than 16-bits (for example, DINT, LINT, REAL, LREAL), multiple Modbus addresses may be required but the most significant byte is always first.

Mapping Address Space and supported Data Types

Since Micro800 uses symbolic variable names instead of physical memory addresses, a mapping from symbolic Variable name to physical Modbus addressing is supported in Connected Components Workbench software, for example, InputSensorA is mapped to Modbus address 100001.

By default Micro800 follows the six-digit addressing specified in the latest Modbus specification. For convenience, conceptually the Modbus address is mapped with the following address ranges. The Connected Components Workbench mapping screen follows this convention.

Variable Data Type	0 - Coils 000001 to 065536		1 - Discrete Inputs 100001 to 165536		3 - Input Registers 300001 to 365536		4 - Holding Registers 400001 to 465536	
	Supported	Modbus Address Used	Supported	Modbus Address Used	Supported	Modbus Address Used	Supported	Modbus Address Used
BOOL	Y	1	Y	1				
SINT	Y	8	Y	8				
BYTE	Y	8	Y	8				
USINT	Y	8	Y	8				
INT	Y	16	Y	16	Y	1	Y	1
UINT	Y	16	Y	16	Y	1	Y	1
WORD	Y	16	Y	16	Y	1	Y	1
REAL	Y	32	Y	32	Y	2	Y	2
DINT	Y	32	Y	32	Y	2	Y	2
UDINT	Y	32	Y	32	Y	2	Y	2
DWORD	Y	32	Y	32	Y	2	Y	2
LWORD	Y	64	Y	64	Y	4	Y	4
ULINT	Y	64	Y	64	Y	4	Y	4
LINT	Y	64	Y	64	Y	4	Y	4
LREAL	Y	64	Y	64	Y	4	Y	4

NOTE: Strings are not supported.

In order to make it easier to map variables to five-digit Modbus addresses, the Connected Components Workbench mapping tool checks the number of characters entered for the Modbus Address. If only five-digits are entered, the address is treated as a five-digit Modbus address. This means that the Coils are mapped from 00001...09999, Discrete Inputs are mapped from 10001...19999, Input Registers are mapped from 30001...39999, and Holding Registers are mapping from 40001...49999.

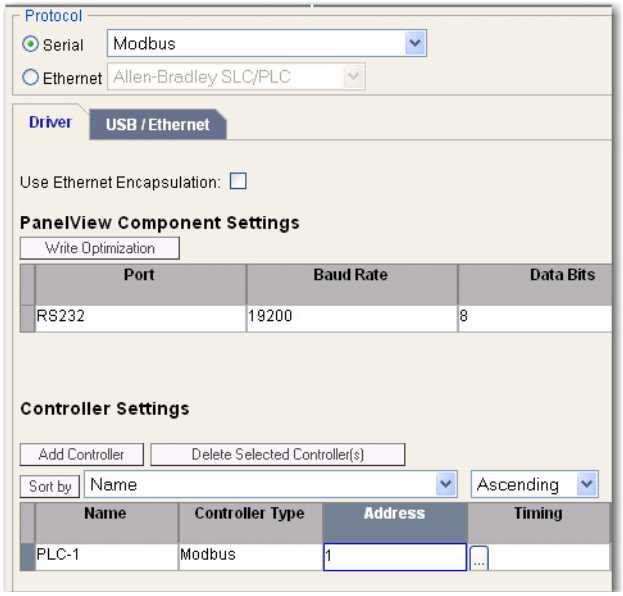
Example 1, PanelView Component HMI (Master) to Micro800 (Slave)

The embedded serial port is targeted for use with HMIs using Modbus RTU. The maximum recommended cable distance is 3 meters. Use the 2080-SERIALISOL serial port plug-in module if longer distances or more noise immunity is needed.

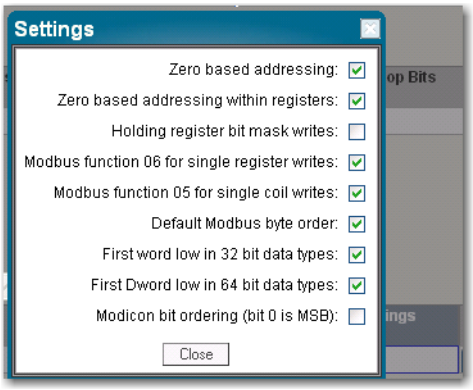
The HMI is typically configured for Master and the Micro800 embedded serial port is configured for Slave.

From the default Communications Settings for a PanelView Component HMI (PVC), there are three items that must be checked or modified in order to set up communications from PVC to Micro800.

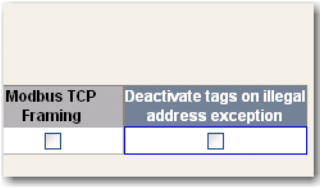
1. Change from DF1 to Modbus protocol.



2. Set the Address of Micro800 slave to match the serial port configuration for the controller.


Settings
Zero based addressing:
Zero based addressing within registers:
Holding register bit mask writes:
Modbus function 06 for single register writes:
Modbus function 05 for single coil writes:
Default Modbus byte order:
First word low in 32 bit data types:
First Dword low in 64 bit data types:
Modicon bit ordering (bit 0 is MSB):
Close

3. Deactivate Tags on Error. This is to prevent the requirement of power cycling PVC when new Modbus Mappings are downloaded from Connected Components Workbench to Micro800 controller.



Example 2, Micro800 (Master) to PowerFlex 4M Drive (Slave)

The following is the overview of the steps to be taken for configuring a PowerFlex 4M drive.

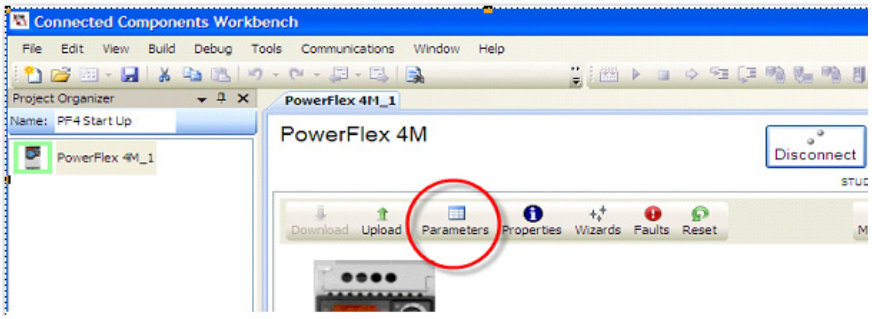
Parameter numbers listed in this section are for a PowerFlex 4M and will be different if you are using another PowerFlex 4-Class drive.

Parameter Name	Parameter Number						
	4M	4	40	40P	400	400N	400P
Start Source	P106	P36					
Speed Reference	P108	P38					
Comm Data Rate	C302	A103			C103		
Comm Node Addr	C303	A104			C104		
Comm Loss Action	C304	A105			C105		
Comm Loss Time	C305	A106			C106		
Comm Format	C306	A107			C102		

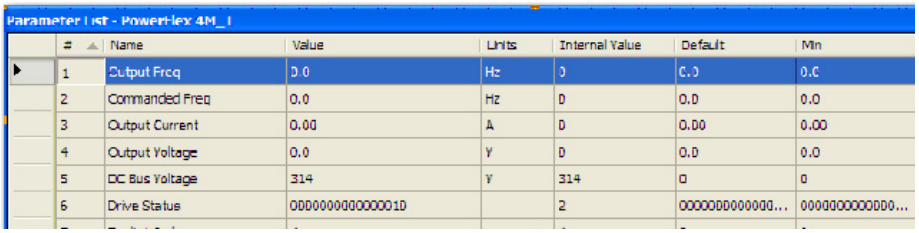
- Connect the 1203-USB to the PowerFlex Drive and to the Computer.
- Launch Connected Components Workbench, Connect to the Drive and set parameters.

To configure PowerFlex 4M, perform the following steps:

1. Double-click the PowerFlex 4M if it is not already open in Connected Components Workbench.
2. Click Connect.
3. In the Connection Browser, expand the AB_DF1 DH+ Driver. Select the AB DSI (PF4 Port) and click OK.
4. Once the Drive has connected and been read in, select the Start up wizard and change the following items. Select Finish to save the changes to the drive.
   - Select the Comm Port as the Speed Reference. Set P108 [Speed Reference] to 5 (Comm Port).
   - Set Start Source to Comm Port. Set P106 [Start Source] to 5 (Comm Port).
   - Defaults for the remaining Inputs
   - Accept Defaults for the remainder and click Finish.
5. Select Parameters from the Connected Components Workbench window.



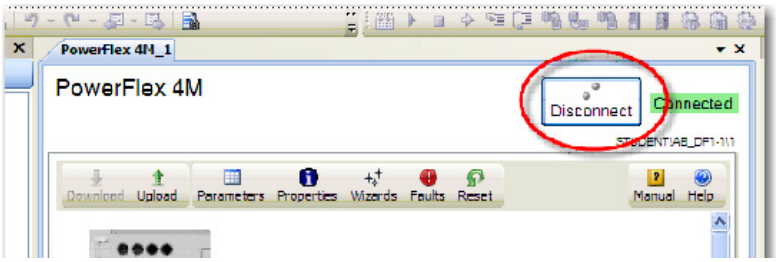
6. The Parameter window opens. Resize it to view the parameters. From this window, you can view and set data values of Parameters.


Parameter List - PowerFlex 4M_1
Name Value Units Internal Value Default Min
1 Output Freq 0.0 Hz 0 0.0 0.0
2 Commanded Freq 0.0 Hz 0 0.0 0.0
3 Output Current 0.00 A 0 0.00 0.00
4 Output Voltage 0.0 V 0 0.0 0.0
5 DC Bus Voltage 314 V 314 0 0
6 Drive Status 0000000000000010 2

7. From the Parameter window, change the following Parameters to set the communications for Modbus RTU so that the PowerFlex 4M Drive will communicate with Micro830/850 via Modbus RTU communication.

Parameter	Description	Setting
C302	Comm. Data Rate (Baud Rate) 4 = 19200 bps	4
C303	Communication Node Address (address range is 1…127)	2
C304	Comm. Loss Action (Action taken when loss communication) 0 = Fault with coast stop	0
C305	Comm. Loss Time (Time remain in communication before taking action set in C304) 5 sec (Max. 60)	5
C306	Comm. Format (Data/Parity/Stop) RTU:8 Data Bit, Parity None, 1 Stop bit	0

8. Disconnect the Communications and save your project.



9. Turn off the power to the drive until the PowerFlex 4M display blanks out completely, then restore power to the PowerFlex 4M.
The drive is now ready to be controlled by Modbus RTU communication commands initiated from the Micro830/850 controller.

Modbus devices can be 0-based (registers are numbered starting at 0), or 1-based (registers are numbered starting at 1). When PowerFlex 4-Class drives are used with Micro800 family controllers, the register addresses listed in the PowerFlex User Manuals need to be offset by n+1.

For example, the Logic Command word is located at address 8192, but your Micro800 program needs to use 8193 (8192+1) to access it.

Modbus Address (n+1 value shown)
8193 Logic Command word (Stop, Start, Jog, etc.)

8194	Speed Reference word xxx.x format for 4/4M/40, where "123" = 12.3 Hz xxx.xx format for 40P/400/400N/400P, where "123" = 1.23 Hz
8449	Logic Status word (Read, Active, Fault, and so on.)
8452	Speed Feedback word (uses same format as Speed Reference)
8450	Error Code word
(n+1)	To access Parameter 'n'

TIP

- If the respective PowerFlex drive supports Modbus Function Code 16 Preset (Write) Multiple Registers, use a single write message with a length of "2" to write the Logic Command (8193) and Speed reference (8194) at the same time.
- Use a single Function Code 03 Read Holding Registers with a length of "4" to read the Logic status (8449), Error Code (8450), and Speed Feedback (8452) at the same time.

Refer to the respective PowerFlex 4-Class drive User Manual for additional information about Modbus addressing. (See Appendix E – Modbus RTU Protocol, on publication 22C-UM001G).

Performance

The performance of MSG_MODBUS (Micro800 is master) is affected by the Program Scan because messages are serviced when the message instruction is executed in a program. For example, if the program scan is 100 ms and six serial ports are used, then the theoretical maximum for serial ports is 60 messages/second total. This theoretical maximum may not be possible since MSG_MODBUS is a master/slave request/response protocol, so performance is affected by several variables such as message size, baud rate, and slave response time.

The performance of Micro800 when receiving Modbus request messages (Micro800 is slave) is also affected by the Program Scan. Each serial port is serviced only once per program scan.

Quickstarts

This chapter covers some common tasks and quickstart instructions that are aimed to make you familiar with the in Connected Component Workbench. The following quickstarts are included:

Topic	Page
Flash Upgrade Your Micro800 Firmware	201
Establish Communications Between RSLinx and a Micro830/Micro850 Controller through USB	204
Configure Controller Password	210
Use the High Speed Counter	214
Forcing I/Os	227
Using Run Mode Change	229

Flash Upgrade Your Micro800 Firmware

This quick start will show you how to flash update the firmware for a Micro800 controller using Connected Components Workbench software release 10 or later.

From Connected Components Workbench software release 10 onwards, there are two options you can select when flash updating the firmware:

- Upgrade or Downgrade – This option retains the controller's existing configuration, Ethernet settings, and password.
- Reset – This option clears the controller's existing configuration, Ethernet settings, and password.

The procedure to flash update the controller is similar for both options.

ATTENTION: Retention of the controller's existing configuration, Ethernet settings, and password is only available when flash updating from firmware revision 10 to the same or later revision. If updating from firmware revision 10 to 9 or earlier, or updating to firmware revision 10 from an earlier revision, the controller's existing configuration, Ethernet settings, and password will be cleared.

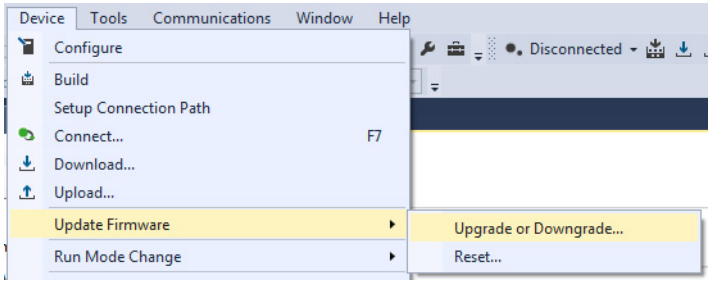
IMPORTANT If you have forgotten the password for the controller, use the Reset option to clear the password.

On Micro850 controllers, users can use flash update their controllers through the Ethernet port, in addition to the USB.

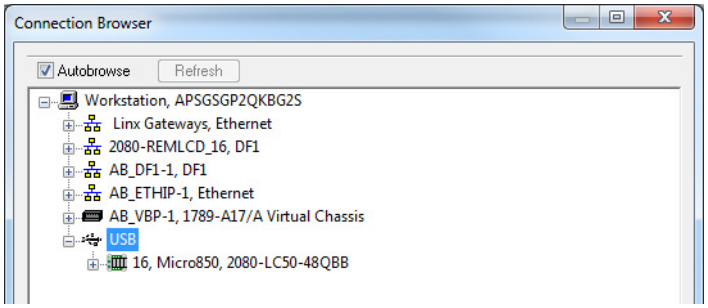
IMPORTANT To successfully flash update your controller over USB, connect only one controller to your computer, and do not perform the flash update in a virtual machine such as VMware.

To begin, launch the Connected Components Workbench software:

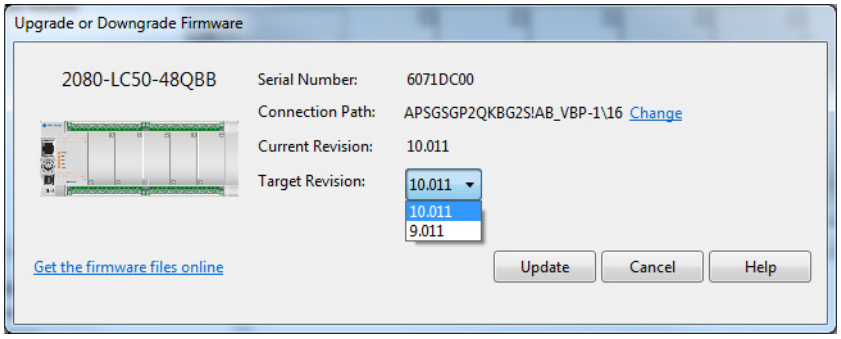
1. In the menu, select Device -> Update Firmware -> Upgrade or Downgrade...
   Alternatively, in the Project Organizer, right-click the controller and select Update Firmware -> Upgrade or Downgrade...



2. If your project does not have a connection path to the controller, the Connection Browser dialog appears. Select your controller, then click OK.



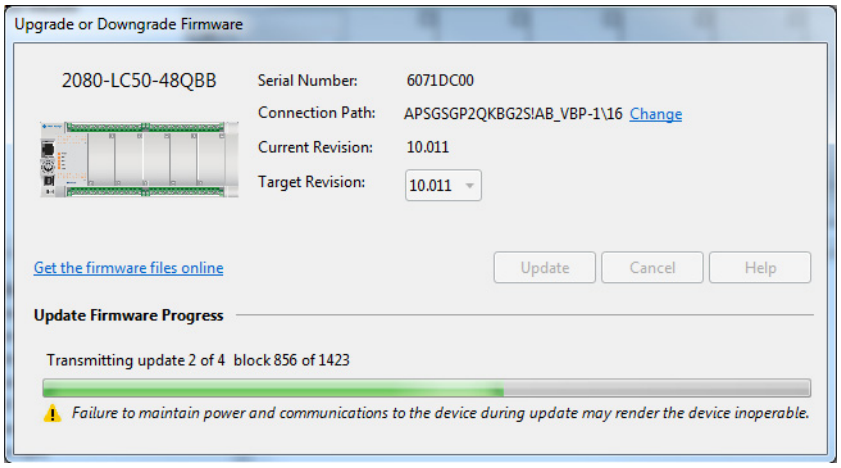
3. In the Upgrade or Downgrade Firmware dialog box, select the desired Target Revision to flash update the controller.



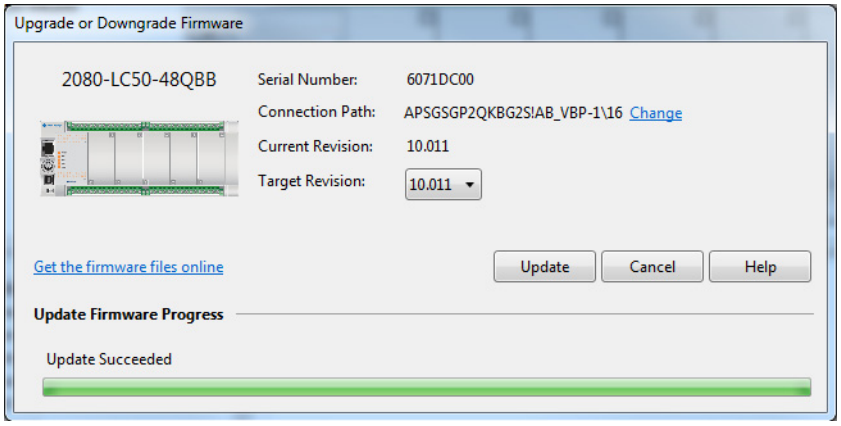
If the desired firmware revision is not shown in the drop-down list, you can download that firmware revision by clicking the “Get the firmware files online” link.

You can also change the Connection Path by clicking the “Change” link.

4. When you have confirmed the settings, click Update to begin flash updating the controller.
   The update progress is shown in the dialog box.


Upgrade or Downgrade Firmware
2080-LC50-48QBB
Serial Number: 6071DC00
Connection Path: APSGSGP2QKBG2S!AB_VBP-1\16 Change
Current Revision: 10.011
Target Revision: 10.011
Get the firmware files online
Update
Cancel
Help
Update Firmware Progress
Transmitting update 2 of 4 block 856 of 1423
Failure to maintain power and communications to the device during update may render the device inoperable.

5. After the update is completed, the status is shown in the dialog box.



Establish Communications Between RSLinx and a Micro830/Micro850 Controller through USB

This quick start shows you how to get RSLinx RSWho to communicate with a Micro830 or Micro850 controller through USB. Micro830 and Micro850 controllers use the AB_VBP-x driver.

RSLinx Classic is installed as part of the Connected Components Workbench software installation process. The minimum version of RSLinx Classic with full Micro800 controller support is 2.57, build 15 (released March 2011).

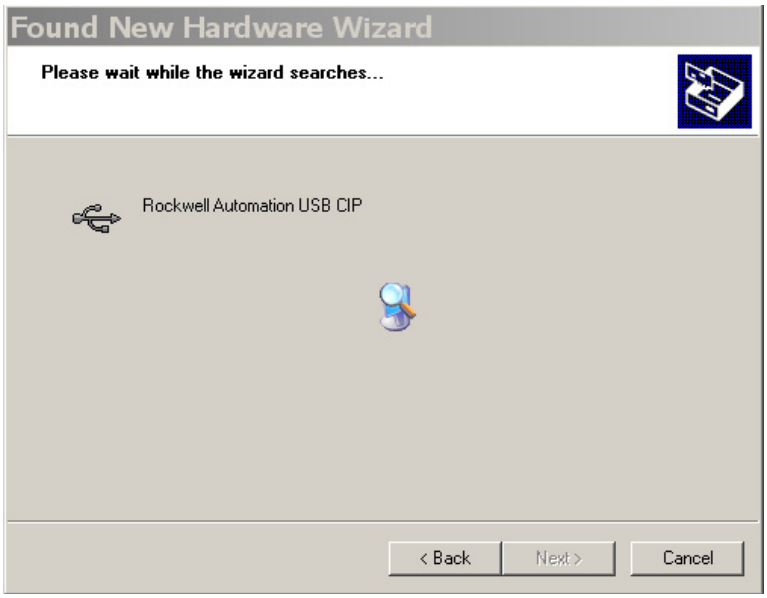
1. Power up the Micro830/Micro850 controller.
2. Plug USB A/B cable directly between your PC and the Micro830/ Micro850 controller.
3. Windows should discover the new hardware. Click No, not this time and then click Next.

Found New Hardware Wizard
Welcome to the Found New Hardware Wizard
Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).
Read our privacy policy
Can Windows connect to Windows Update to search for software?
Yes, this time only
Yes, now and every time I connect a device
No, not this time
Click Next to continue.
< Back
Next >
Cancel

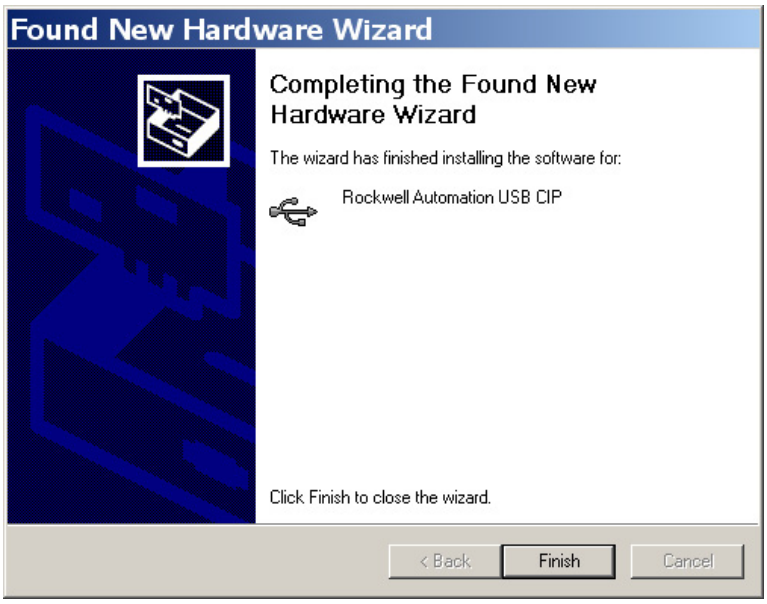
4. Click Install the software automatically (Recommended), and then click Next.

Found New Hardware Wizard
This wizard helps you install software for:
Rockwell Automation USB CIP
If your hardware came with an installation CD or floppy disk, insert it now.
What do you want the wizard to do?
Install the software automatically (Recommended)
Install from a list or specific location (Advanced)
Click Next to continue.
< Back
Next >
Cancel

The Wizard searches for new hardware.

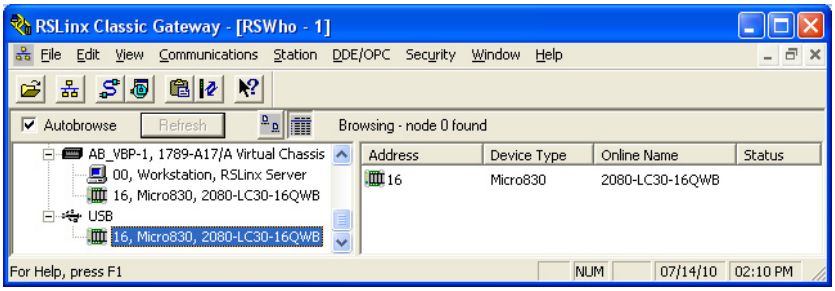


5. Click Finish when the wizard completes the installation.

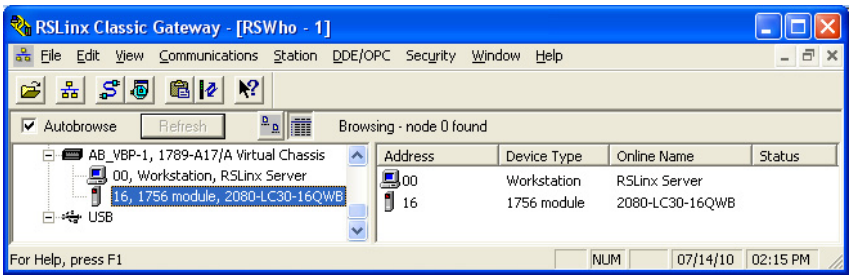

Found New Hardware Wizard
Completing the Found New Hardware Wizard
The wizard has finished installing the software for:
Rockwell Automation USB CIP
Click Finish to close the wizard.
< Back
Finish
Cancel

6. Open RSLinx Classic and run RSWho by clicking the icon.

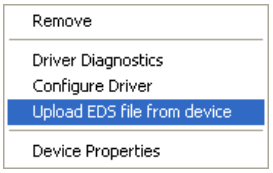
   If the proper EDS file is installed, the Micro830/Micro850 controller should be properly identified and show up under both the Virtual Backplane (VBP) driver and the USB driver, which was automatically created.



   If instead the Micro830/Micro850 shows up as a "1756 Module" under the AB_VBP-1 Virtual Chassis driver, then the proper EDS file for this major revision of firmware has not yet been installed or the controller is running pre-release firmware (Major Revision=0).



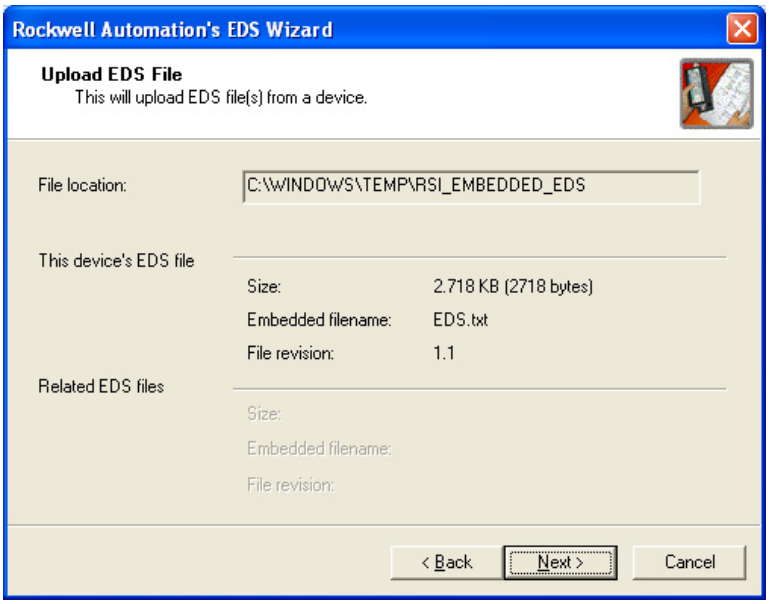
Since Micro830/Micro850 controllers support embedded EDS files, right click this device and select Upload EDS file from device.

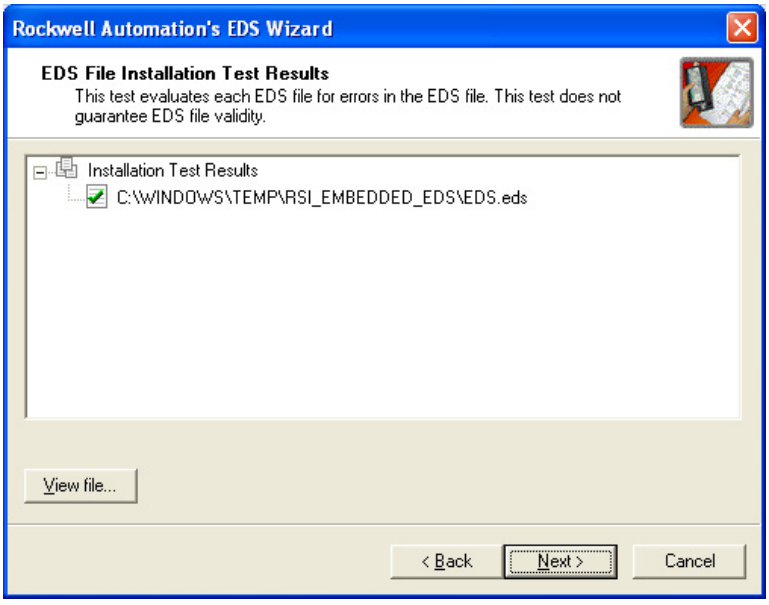


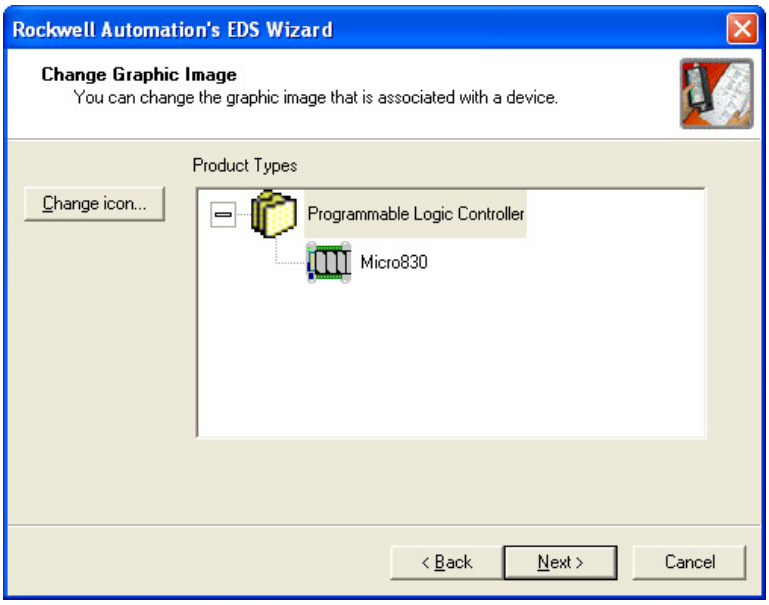
7. On the EDS wizard that appears , click Next to continue.

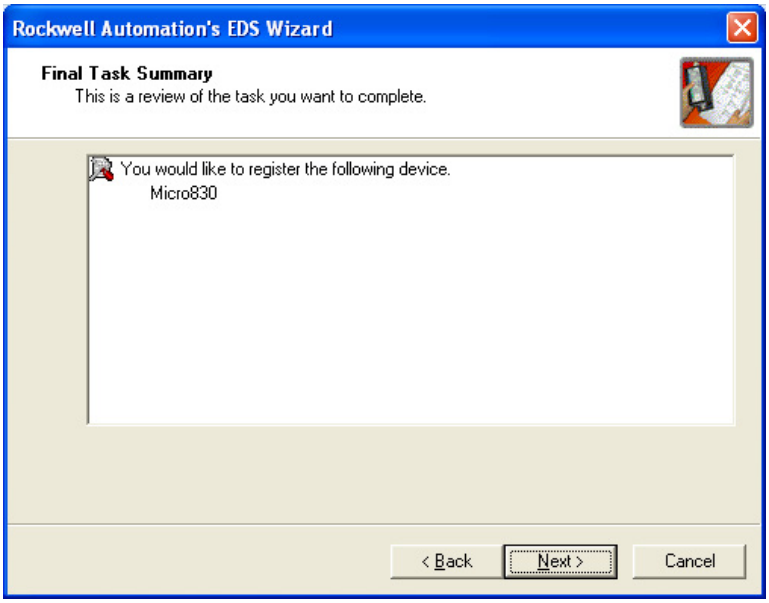
Rockwell Automation's EDS Wizard
Welcome to Rockwell Automation's EDS Wizard
The EDS Wizard allows you to:
- register EDS-based devices.
- unregister a device.
- change the graphic images associated with a device.
- create an EDS file from an unknown device.
- upload EDS file(s) stored in a device.
To continue click Next
Next >
Cancel

8. Follow the prompts to upload and install the EDS file.

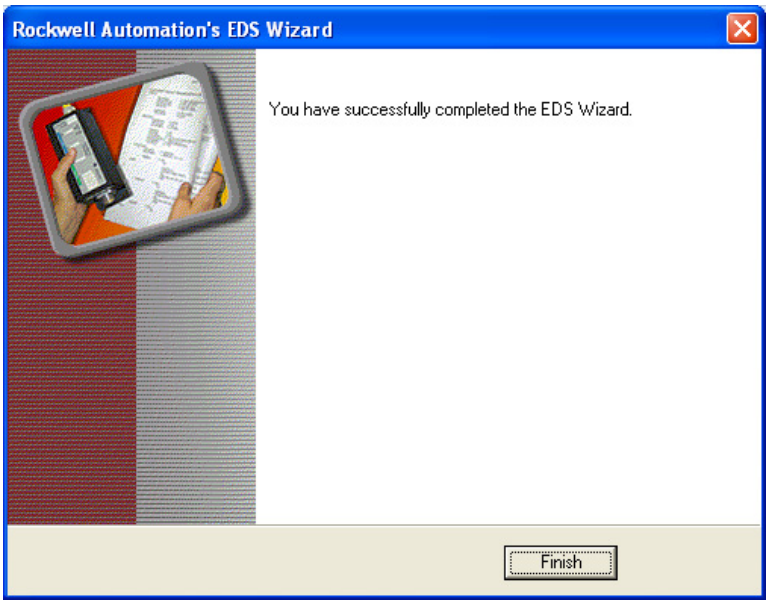



Rockwell Automation's EDS Wizard
EDS File Installation Test Results
This test evaluates each EDS file for errors in the EDS file. This test does not guarantee EDS file validity.
Installation Test Results
C:\WINDOWS\TEMP\RSI_EMBEDDED_EDS\EDS.eds
View file...
< Back
Next >
Cancel


Rockwell Automation's EDS Wizard
Change Graphic Image
You can change the graphic image that is associated with a device.
Product Types
Change icon...
Programmable Logic Controller
Micro830
< Back
Next >
Cancel


Rockwell Automation's EDS Wizard
Final Task Summary
This is a review of the task you want to complete.
You would like to register the following device.
Micro830
< Back
Next >
Cancel

9. Click Finish to complete.



If the Micro830/Micro850 still shows up as a 1756 Module, then you are probably running pre-release firmware which is reporting itself as Major Revision 0, which does not match the embedded EDS file. To confirm, right click the device and select Device Properties (firmware Revision is Major.Minor).

Configure Controller Password

Set, change, and clear the password on a target controller through the Connected Components Workbench software.

IMPORTANT The following instructions are supported on Connected Components Workbench revision 2 and Micro800 controllers with firmware revision 2.

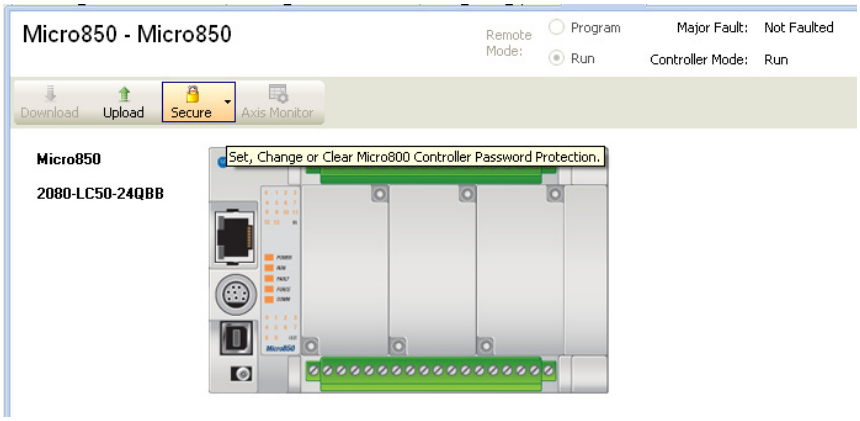
For more information about the controller password feature on Micro800 controllers, see Controller Security on page 161.

Set Controller Password

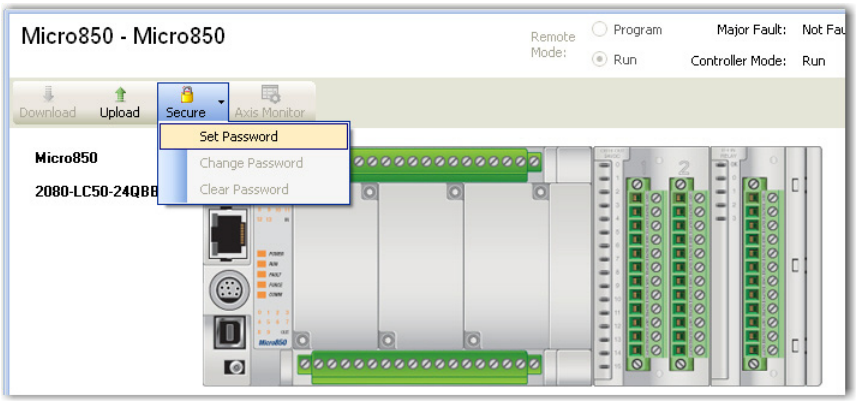
IMPORTANT After creating or changing the controller password, you need to power down the controller in order for the password to be saved.

In the following instructions, the Connected Components Workbench software is connected to the Micro800 controller.

1. On the Connected Components Workbench software, open the project for the target controller.
2. Click Connect to connect to the target controller.
   On the Device Details toolbar, roll over the Secure button. The tooltip message “Set, Change, or Clear Micro800 Controller Password Protection” is displayed.



3. Click Secure button. Select Set Password.



4. The Set Controller Password dialog appears. Provide password. Confirm the password by providing it again in the Confirm field.

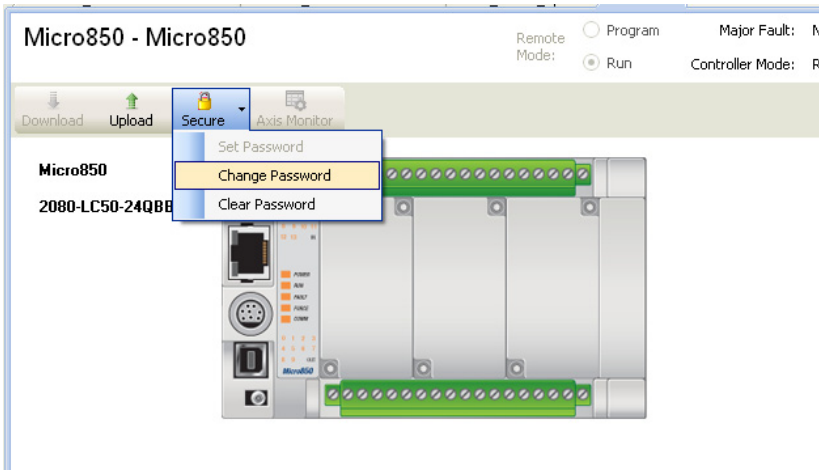
TIP Passwords must have at least eight characters to be valid.

5. Click OK.
   Once a password is created, any new sessions that try to connect to the controller will have to supply the password to gain exclusive access to the target controller.

Change Password

With an authorized session, you can change the password on a target controller through the Connected Components Workbench software. The target controller must be in Connected status.

1. On the Device Details toolbar, click Secure button. Select Change Password.



2. The Change Controller Password dialog appears. Enter Old Password, New Password and confirm the new password.

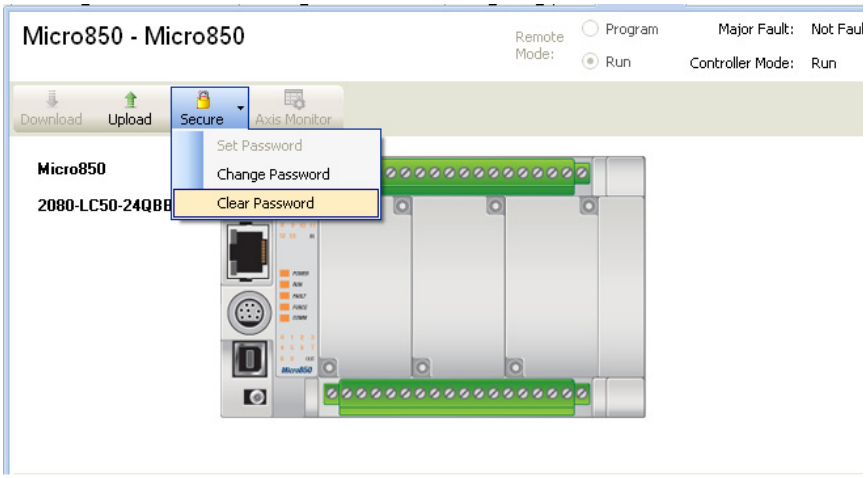
3. Click OK.

The controller requires the new password to grant access to any new session.

Clear Password

With an authorized session, you can clear the password on a target controller through the Connected Components Workbench software.

1. On the Device Details toolbar, click Secure button. Select Clear Password.



2. The Clear Password dialog appears. Enter Password.
3. Click OK to clear the password.

The controller will require no password on any new session.

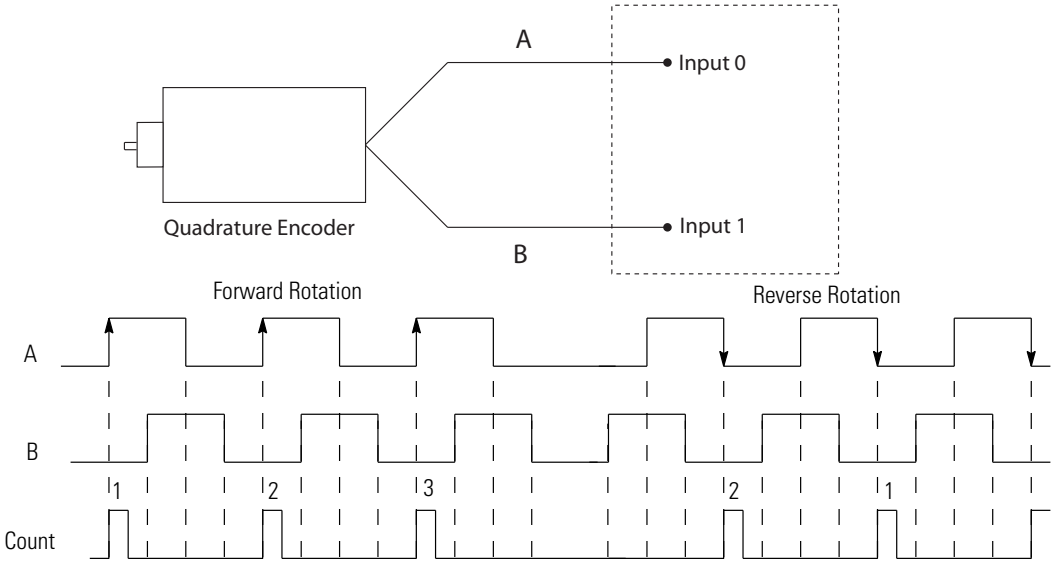
Use the High Speed Counter

To use HSC, you first need to establish the HSC counting mode required by your application. See HSC Mode (HSCAPP.HSCMode) on page 134 for available modes on Micro800 controllers.

The following sample project guides you through the creation of a project which uses HSC mode 6, a quadrature counter with phased inputs A and B. It shows you how to write a simple ladder program with the HSC function block, create variables, and assign variables and values to your function block. You will also be guided through a step-by-step process on how test your program, and enable a Programmable Light Switch (PLS).

This sample project makes use of a quadrature encoder. The quadrature encoder is used for determining direction of rotation and position for rotating, such as a lathe. The Bidirectional Counter counts the rotation of the Quadrature Encoder.

The figure below shows a quadrature encoder connected to inputs 0 and 1. The count direction is determined by the phase angle between A and B. If A leads B, the counter increments. If B leads A, the counter decrements.

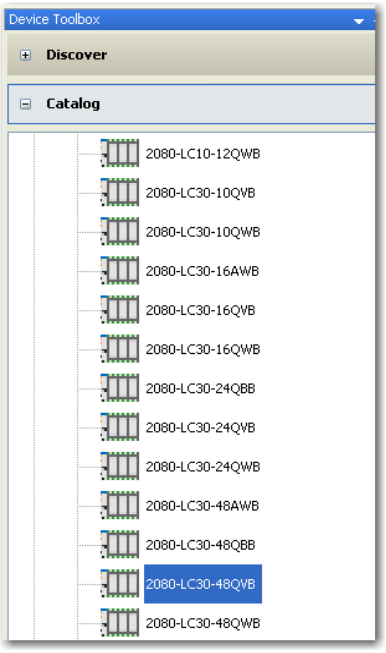


This quickstart includes the following sections:

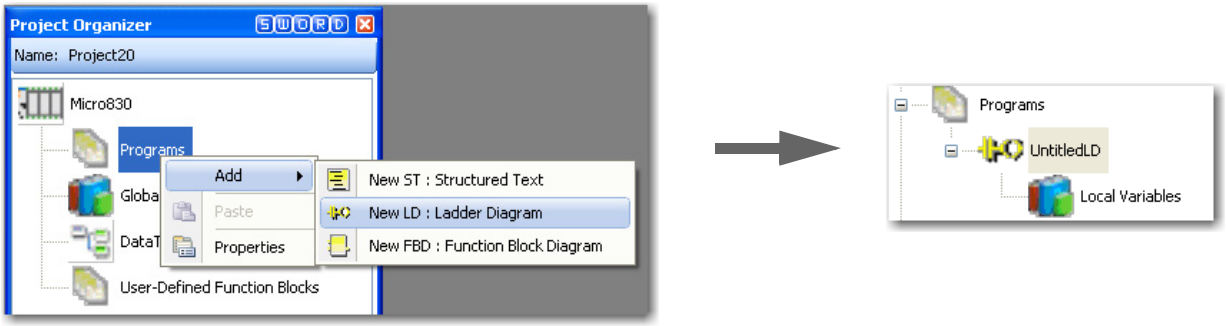
- Create the HSC Project and Variables on page 216
- Assign Values to the HSC Variables on page 219
- Assign Variables to the Function Block on page 222
- Run the High Speed Counter on page 223
- Use the Programmable Limit Switch (PLS) Function on page 225

Create the HSC Project and Variables

1. Start Connected Components Workbench and open a new project. From the Device Toolbox, go to Catalog → Controllers. Double-click your controller[1] or drag and drop it onto the Project Organizer windows.

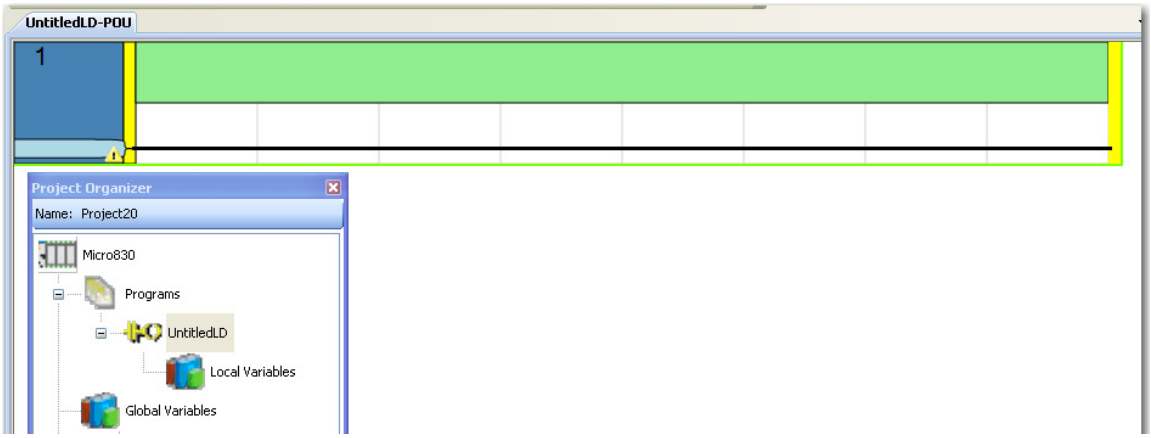


2. Under Project Organizer, right-click Programs. Click Add New LD: Ladder Diagram to add a new ladder logic program.

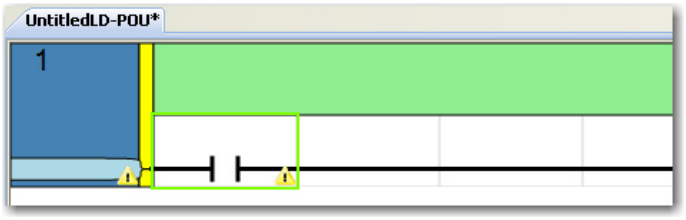


(1) The HSC is supported on all Micro830 and Micro850 controllers, except on 2080-LCxx-xxAWB types.

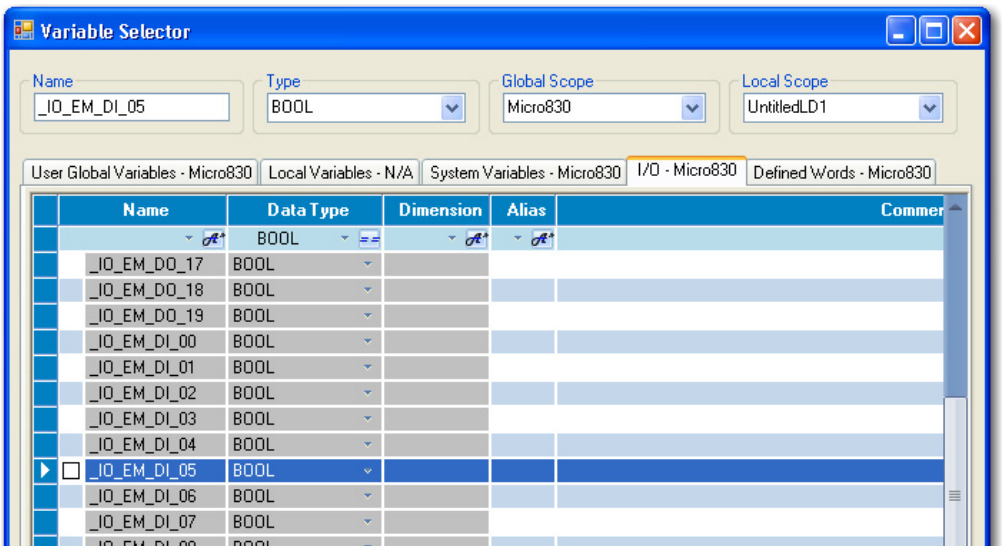
3. Right-click UntitledLD and select Open.



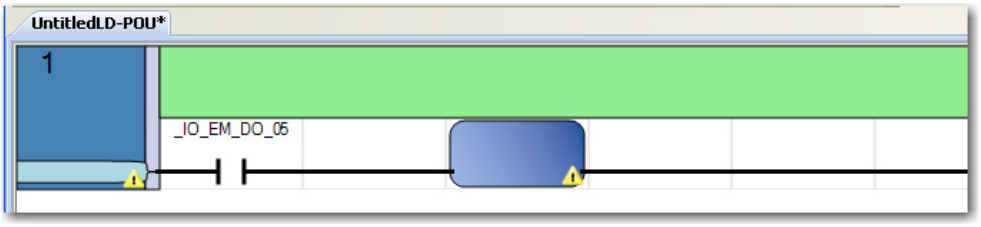
4. From the Toolbox, double-click Direct Contact to add it to the rung or drag and drop Direct Contact onto the Rung.



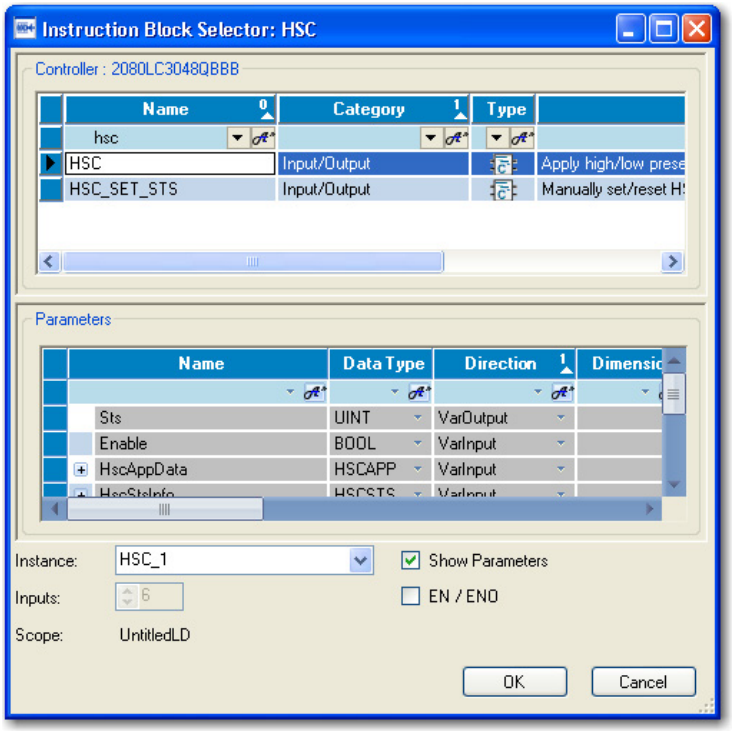
5. Double-click the Direct Contact you have just added to bring up the Variable Selector dialog. Click I/O Micro830 tab. Assign the Direct Contact to input 5 by selecting _IO_EM_DI_05. Click OK.



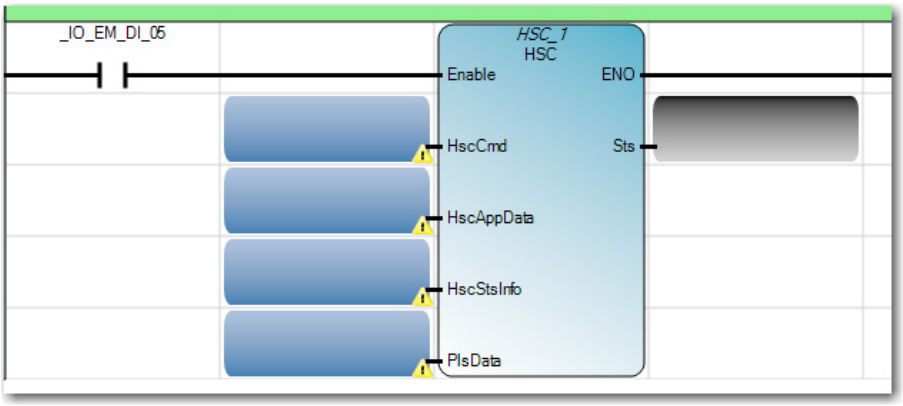
6. To the right of the Direct Contact, add a function block by double-clicking function block from the Toolbox or dragging and dropping the function block onto the rung.



7. Double-click the function block to open up Instruction Selector dialog. Choose HSC. You can do a quick search for HSC function block by typing "hsc" on the name field. Click OK.



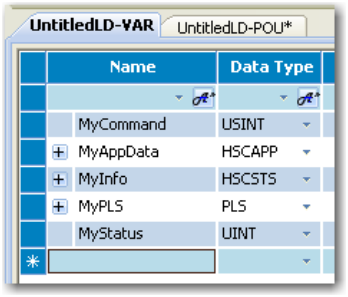
Your ladder rung should appear as shown below:



8. On the Project Organizer pane, double-click Local Variables to bring up the Variables window. Add the following variables with the corresponding data types, as specified in the table.

Variable Name	Data Type
MyCommand	USINT
MyAppData	HSCAPP
MyInfo	HSCSTS
MyPLS	PLS
MyStatus	UINT

After adding the variables, your Local Variables table should look like this:

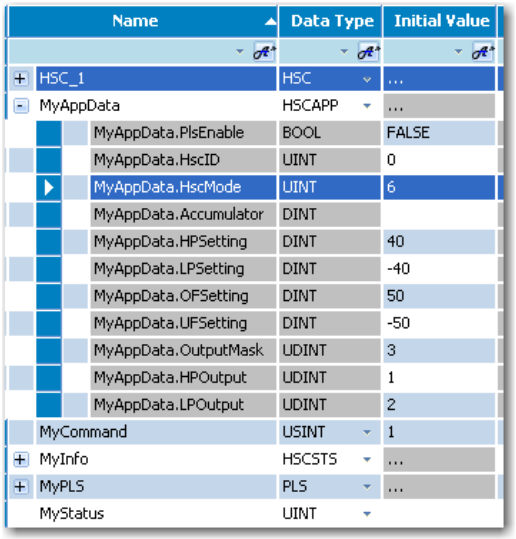


Assign Values to the HSC Variables

Next, you need to assign values to the variables you have just created. Typically, a routine is used to assign values to your variables. For illustration purposes, this quickstart assigns values through the Initial Value column of the Local Variables table.

TIP In a real program, you should write a routine to assign values to your variable according to your application.

1. On the Initial Value field for the MyCommand variable, type 1. See HSC Commands (HScCmd) on page 151 for more information on the description for each value.
2. Assign values to the MyAppData variables. Expand the list of MyAppData sub-variables clicking the + sign. Set the values of the different sub-variables as shown in the following screenshot.



Name	Data Type	Initial Value
HSC_1	HSC	...
MyAppData	HSCAPP	...
MyAppData.PlsEnable	BOOL	FALSE
MyAppData.HscID	UINT	0
MyAppData.HscMode	UINT	6
MyAppData.Accumulator	DINT	
MyAppData.HPSetting	DINT	40
MyAppData.LPSetting	DINT	-40
MyAppData.OFSetting	DINT	50
MyAppData.UFSetting	DINT	-50
MyAppData.OutputMask	UDINT	3
MyAppData.HPOutput	UDINT	1
MyAppData.LPOutput	UDINT	2
MyCommand	USINT	1
MyInfo	HSCSTS	...
MyPLS	PLS	...
MyStatus	UINT	

IMPORTANT MyAppData variable has sub-variables which determine the settings of the counter. It is **crucial** to know each one in order to determine how the counter will perform. A quick summary is provided below but you can also see HSC APP Data Structure on page 133 for detailed information.

MyAppData.PlsEnable allows the user to either enable or disable the PLS settings. It should be set to FALSE (disabled) if the MyAppData variable is to be used.

MyAppData.HscID allows the user to specify which embedded inputs will be used depending on the mode and the type of application. See the table HSC Inputs and Wiring Mapping on page 129 to know the different IDs that can be used as well as the embedded inputs and its characteristics.

If ID 0 is used, ID 1 cannot be used on the same controller since the inputs are being used by the Reset and Hold.

MyAppData.HscMode allows the user to specify the type of operation in which the HSC will use to count. See HSC Mode (HSCAPP.HSCMode)

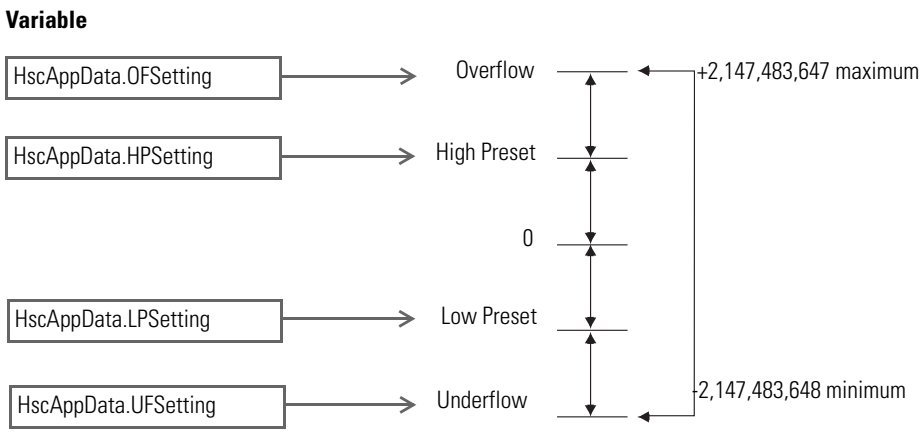
on page 134 for more information about HSC modes. You can also quickly refer to the table below for the list of ten available modes.

HSC Operating Modes

Mode Number	Type
0	Up Counter – The accumulator is immediately cleared (0) when it reaches the high preset. A low preset cannot be defined in this mode.
1	Up Counter with external reset and hold – The accumulator is immediately cleared (0) when it reaches the high preset. A low preset cannot be defined in this mode.
2	Counter with external direction
3	Counter with external direction, reset, and hold
4	Two input counter (up and down)
5	Two input counter (up and down) with external reset and hold
6	Quadrature counter (phased inputs A and B)
7	Quadrature counter (phased inputs A and B) with external reset and hold
8	Quadrature X4 counter (phased inputs A and B)
9	Quadrature X4 counter (phased inputs A and B) with external reset and hold

Modes 1, 3, 5, 7, and 9 will only work when an ID of 0, 2, or 4 is set due to the fact that these modes use reset and hold. Modes 0, 2, 4, 6, and 8 will work on any ID. Modes 6...9 will only work when an encoder is connected to the controller. Use the HSC ID chart as a reference to wire the encoder to the controller.

MyAppData.HPSetting, **MyAppData.LPSetting**, **MyAppData.OFSetting**, and **MyAppData.UFSetting** are all user-defined variables which represent the counting range of the HSC. The diagram below gives an example of a range of values that can be set for these variables.



MyAppData.OutputMask along with **MyAppData.HPOutput** and **MyAppData.LPOutput** allows the user to specify which embedded

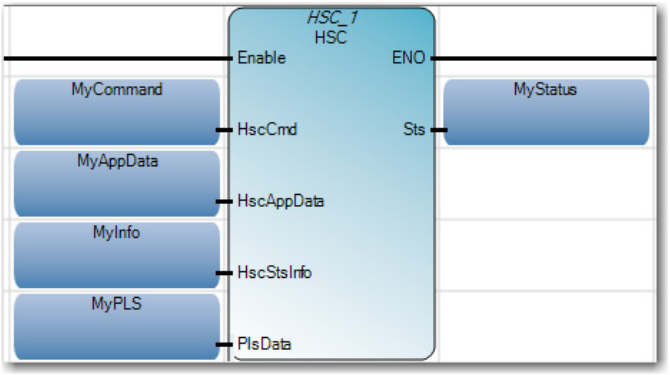
outputs can be turned on when a High Preset or Low Preset is reached. These variables use a combination of decimals and binary numbers to specify the embedded outputs that are able to turn on/off.

Thus, in our example, we first set the Output Mask to a decimal value of 3 which, when converted to binary, is equal to 0011. This means that now outputs O0 and O1 can be turned On/Off.

We have set the HPOutput to a decimal value of 1, which, when converted to binary, is equal to 0001. This means that when a High Preset is reached, output O0 will turn on and stay on until the HSC is reset or the counter counts back down to a Low Preset. The LPOutput works same way as the HPOutput except an output will be turned on when a Low Preset is reached.

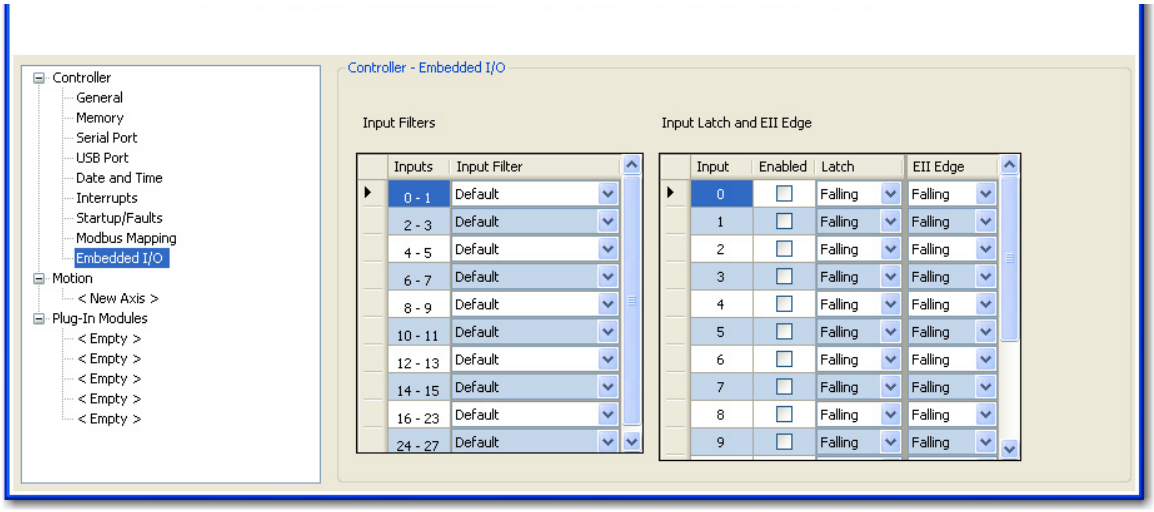
Assign Variables to the Function Block

1. Go back to the ladder diagram and assign the variables you have just configured to the corresponding elements of the HSC function block. The HSC function block should appear as shown in the screenshot:



To assign a variable to a particular element in your function block, double click the empty variable block. On the Variable selector that appears, choose the variable you have just created. (For example, for the input element HSCAppData, select the variable MyAppData.)

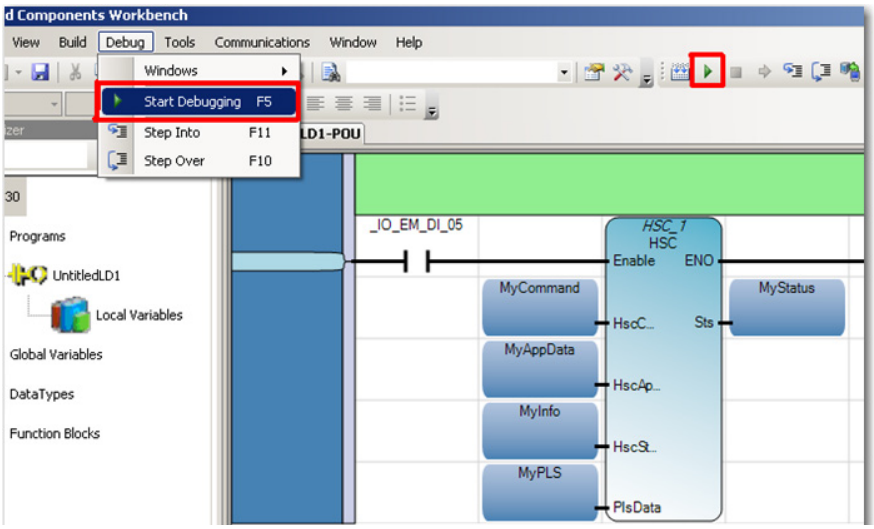
2. Next, click the Micro830 controller under the Project Organizer pane to bring up the Micro830 Controller Properties pane. Under Controller Properties, click Embedded I/O. Set the input filters to a correct value depending on the characteristics of your encoder.



3. Make sure that your encoder is connected to the Micro830 controller.
4. Power up the Micro830 controller and connect it to your PC. Build the program in Connected Components Workbench and download it to the controller.

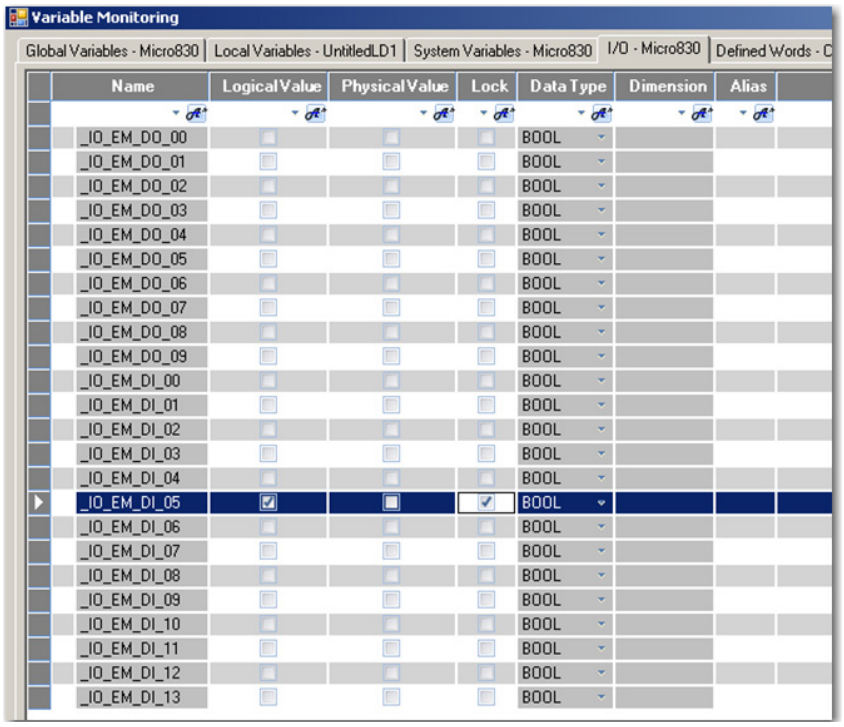
Run the High Speed Counter

1. To test the program, go into debug mode by doing any of the following:
   - Click Debug menu, then choose Start Debugging,
   - Click the green play button below the menu bar, or
   - Hit the F5 windows key.



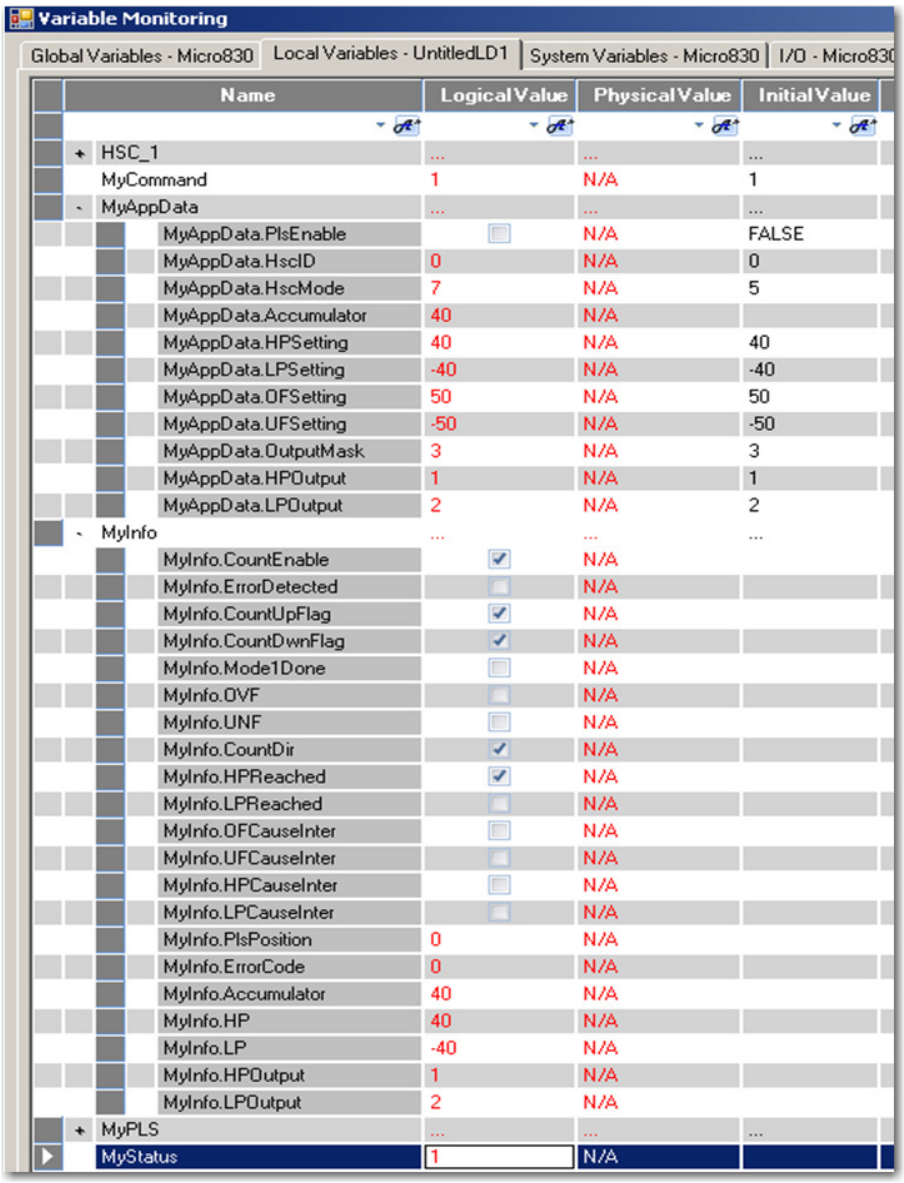
Now that we are on debug mode we can see the values of the HSC output. The HSC function block has two outputs, one is the STS (MyStatus) and the other is the HSCSTS (MyInfo).

2. Double-click the Direct Contact labeled _IO_EM_DI_05 to bring up the Variable Monitoring window.
3. Click the I/O Micro830 tab. Select the _IO_EM_DI_05 row. Check the boxes Lock and Logical Value so that this input will be forced in the ON position.



4. Click the Local Variables tab to see any real time changes being made to the variables. Expand the MyAppData and MyInfo variable list by clicking the + sign.
5. Turn On the encoder to see the counter count up/down. For example, if the encoder is attached to a motor shaft then turn on the motor to trigger the HSC count. The counter value will be displayed on MyInfo.Accumulator. MyStatus variable should display a Logical Value of 1, which means that the HSC is running.

TIP See HSC Function Block Status Codes on page 152 for the complete list of status codes. For example, if the MyStatus value is 04, a configuration error exists and the controller will fault. You need to check your parameters in this case.



For this example, once the Accumulator reaches a High Preset value of 40, output 0 turns on and the HPReached flag turns on. Once the Accumulator reaches a Low Preset value of -40, output 1 turns on and the LPReached flag turns on as well.

Use the Programmable Limit Switch (PLS) Function

The Programmable Limit Switch function allows you to configure the High-Speed Counter to operate as a PLS (programmable limit switch) or rotary cam switch. The PLS is used when you need more than one pair of high and low presets (up to 255 pairs of high and low presets are supported by the PLS).

1. Start a new project following the same steps and values as the previous project. Set the values for the following variables as follows:
   - HSCAPP.PlsEnable variable should be set to TRUE
   - Set a value only for UFSetting and OFSetting (OutputMask is optional depending if an output is to be set or not). Your new values should follow the example below:

UntitledLD1-VAR

Name	Data Type	Dimension	Alias	Initial Value	Attribute
HSC_1	HSC			...	ReadWrite
MyCommand	USINT			1	ReadWrite
MyAppData	HSCAPP			...	ReadWrite
MyAppData.PlsEnable	BOOL			TRUE	ReadWrite
MyAppData.HscID	UINT			0	ReadWrite
MyAppData.HscMode	UINT			7	ReadWrite
MyAppData.Accumulator	DINT				ReadWrite
MyAppData.HPSetting	DINT				ReadWrite
MyAppData.LPSetting	DINT				ReadWrite
MyAppData.OFSetting	DINT			50	ReadWrite
MyAppData.UFSetting	DINT			-50	ReadWrite
MyAppData.OutputMask	UDINT			255	ReadWrite
MyAppData.HPOutput	UDINT				ReadWrite
MyAppData.LPOutput	UDINT				ReadWrite
MyInfo	HSCSTS			...	ReadWrite
MyPLS	PLS	[1..4]		...	ReadWrite
MyPLS[1]	PLS			...	ReadWrite
MyPLS[1].HscHP	DINT			10	ReadWrite
MyPLS[1].HscLP	DINT			-10	ReadWrite
MyPLS[1].HscHPOutPut	UDINT			1	ReadWrite
MyPLS[1].HscLPOutPut	UDINT			16	ReadWrite
MyPLS[2]	PLS			...	ReadWrite
MyPLS[2].HscHP	DINT			20	ReadWrite
MyPLS[2].HscLP	DINT			-20	ReadWrite
MyPLS[2].HscHPOutPut	UDINT			2	ReadWrite
MyPLS[2].HscLPOutPut	UDINT			32	ReadWrite
MyPLS[3]	PLS			...	ReadWrite
MyPLS[3].HscHP	DINT			30	ReadWrite
MyPLS[3].HscLP	DINT			-30	ReadWrite
MyPLS[3].HscHPOutPut	UDINT			4	ReadWrite
MyPLS[3].HscLPOutPut	UDINT			64	ReadWrite
MyPLS[4]	PLS			...	ReadWrite
MyPLS[4].HscHP	DINT			40	ReadWrite
MyPLS[4].HscLP	DINT			-40	ReadWrite
MyPLS[4].HscHPOutPut	UDINT			8	ReadWrite
MyPLS[4].HscLPOutPut	UDINT			128	ReadWrite
MyStatus	UINT				ReadWrite

In this example, the PLS variable is given a dimension of [1..4]. This means that the HSC can have four pairs of High and Low Presets.

Once again, your High Presets should be set lower than the OFSetting and the Low Preset should be greater than the UFSetting. The HscHPOutPut and HscLPOutPut values will determine which outputs will be turned on when a High Preset or Low Preset is reached.

2. You can now build and download the program into the controller then debug and test it following the instructions for the last project.

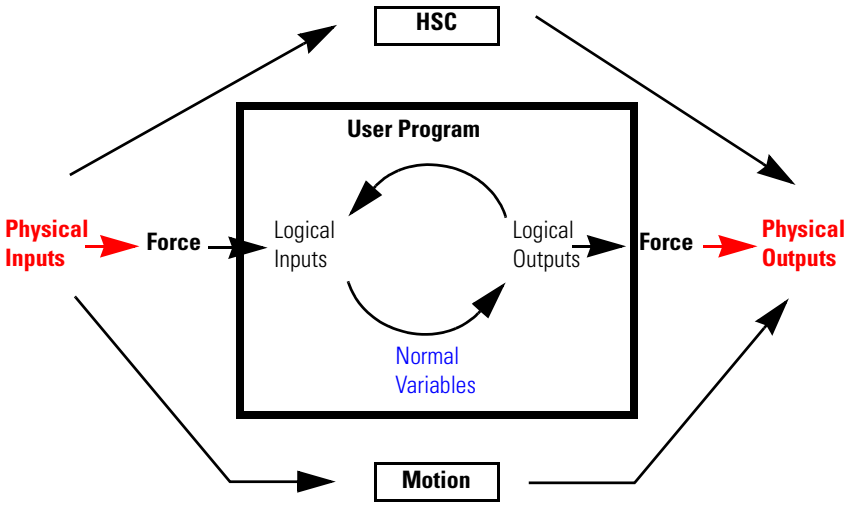
Forcing I/Os

Inputs are logically forced. LED status indicators do not show forced values, but the inputs in the user program are forced.

Forcing is only possible with I/O and does not apply to user defined variables and non-I/O variables, and special functions such as HSC and Motion which execute independently from the User Program scan. For example, for motion, Drive Ready input cannot be forced.

Unlike inputs, outputs are physically forced. LED status indicators do show forced values and the user program does not use forced values.

The following diagram illustrates forcing behavior.



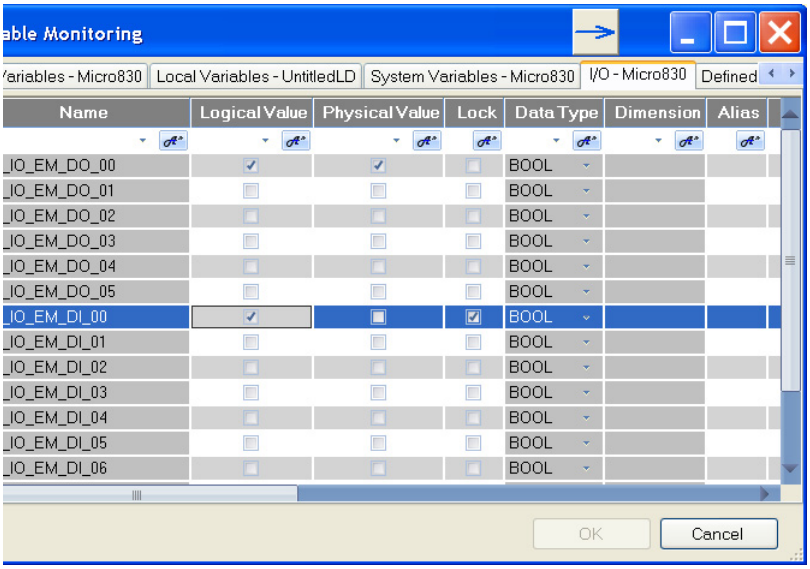
- LED status indicators always match the physical value of I/O
- Normal, non-physical internal variables cannot be forced
- Special functions such as HSC and Motion cannot be forced

ATTENTION: Forcing variable can result in sudden machine movement, possibly injuring personnel or equipment. Use extreme caution when forcing variables.

Checking if Forces (locks) are Enabled

If Connected Components Workbench is available, check the Variable Monitor while debugging online. Forcing is performed by first Locking an I/O variable and then setting the Logical Value for Inputs and Physical Value for Outputs.

Remember you cannot force a Physical Input and cannot force a Logical Output.



In many cases, the front of the controller is not visible to the operator and Connected Components Workbench is not online with the controller. If you want the force status to be visible to the operator, then the User Program must read the force status using the SYS_INFO function block and then display the force status on something that the operator can see, such as the human machine interface (HMI), or stack light. The following is an example program in Structured Text.

```
(* Read System Information including Force Enable bit *)
SYS_INFO_1(TRUE);

(* Turn on Warning Light if Forces are Enabled *)
If SYS_INFO_1.Sts.ForcesInstall = TRUE THEN
    _IO_EM_DO_05 := TRUE;
ELSE
    _IO_EM_DO_05 := FALSE;
END_IF;
```

If the front of the controller is visible, and not blocked by the cabinet enclosure, Micro830 and Micro850 controllers have a Force LED indicator.

I/O Forces After a Power Cycle

After a controller is power cycled, all I/O forces are cleared from memory.

Using Run Mode Change

Run Mode Change allows the user to make small changes to the logic of a running project and immediately testing it out on the controller, without having to go into Program mode or disconnecting from the controller.

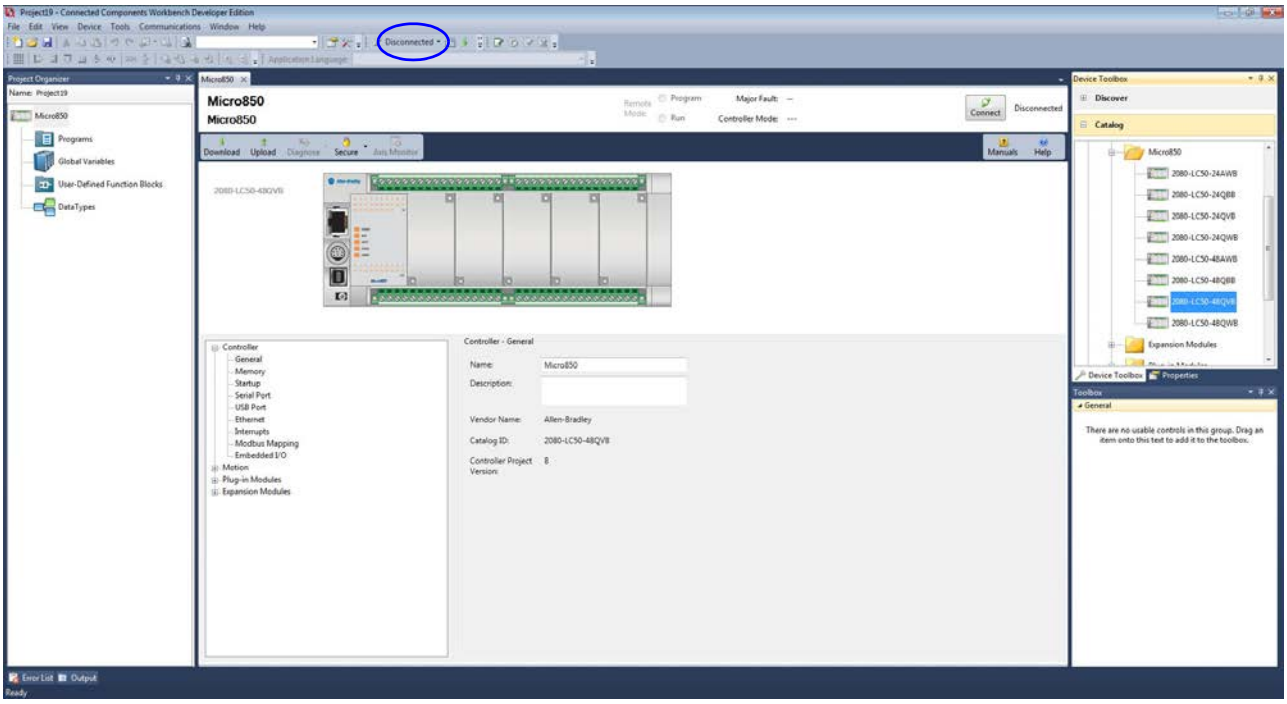
IMPORTANT The following requirements must be met to use Run Mode Change:

- Micro820/Micro830/Micro850 controller firmware revision 8.0 or higher, and
- Connected Components Workbench Developer Edition software, version 8.0 or higher.

The following sample project guides you through the creation of a simple application for a Micro850 controller without any plug-in modules, and how to use the Run Mode Change feature.

Create the Project

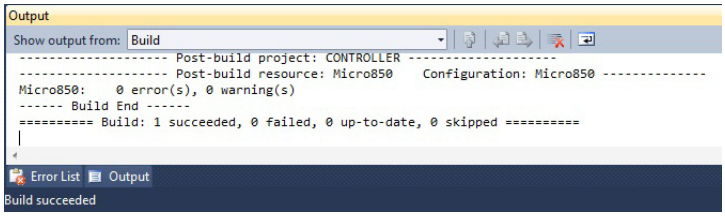
1. Create a new project for a Micro830/Micro850 controller without any plug-ins.
   Observe that the controller is disconnected.



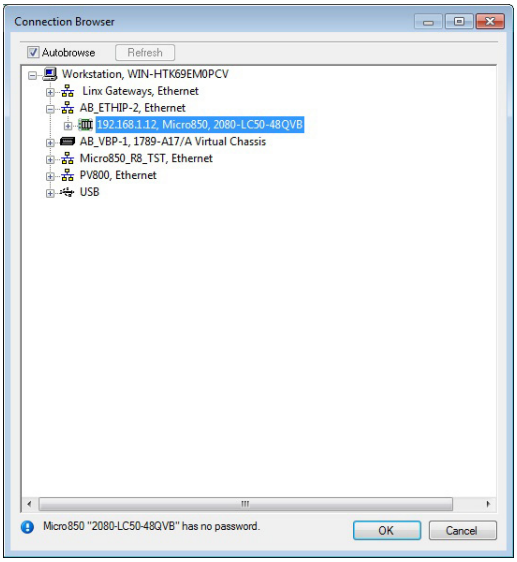
2. Right-click Programs and select Add -> New LD: Ladder Diagram.
3. From the Toolbox, double-click Direct Coil to add it to the rung, or drag and drop Direct Coil onto the rung.

4. Double -click the newly added Direct Coil to bring up the Variable Selector dialog and select "_IO_EM_DO_00".

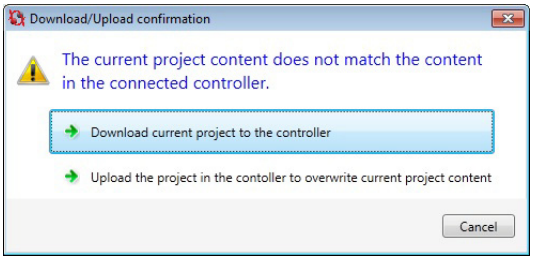
5. Build the project.


Output
Show output from: Build
---------- Post-build project: CONTROLLER ----------
---------- Post-build resource: Micro850 Configuration: Micro850 ----------
Micro850: 0 error(s), 0 warning(s)
------ Build End ------
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
Error List
Output
Build succeeded

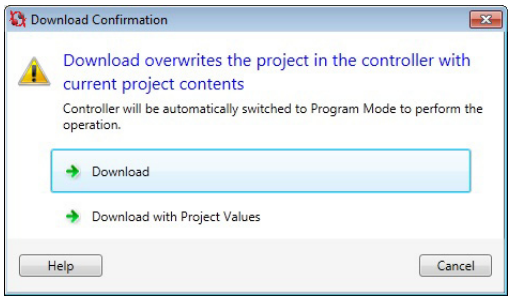
6. Download the project to the controller.
In the Connection Browser dialog, select the Micro850 controller.



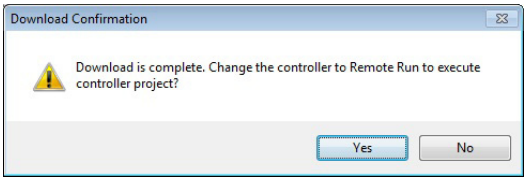
7. Select Download current project to the controller.


Download/Upload confirmation
The current project content does not match the content in the connected controller.
Download current project to the controller
Upload the project in the contoller to overwrite current project content
Cancel

8. Select Download to confirm.


Download Confirmation
Download overwrites the project in the controller with current project contents
Controller will be automatically switched to Program Mode to perform the operation.
Download
Download with Project Values
Help
Cancel

9. When the project has been downloaded to the controller, a prompt asking to change the controller to Remote Run mode appears. Click Yes.

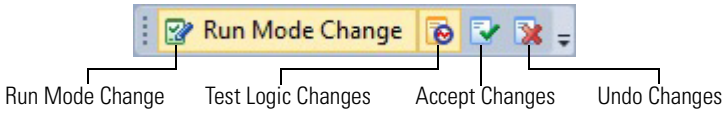


10. Observe that the controller is now in Debug mode.

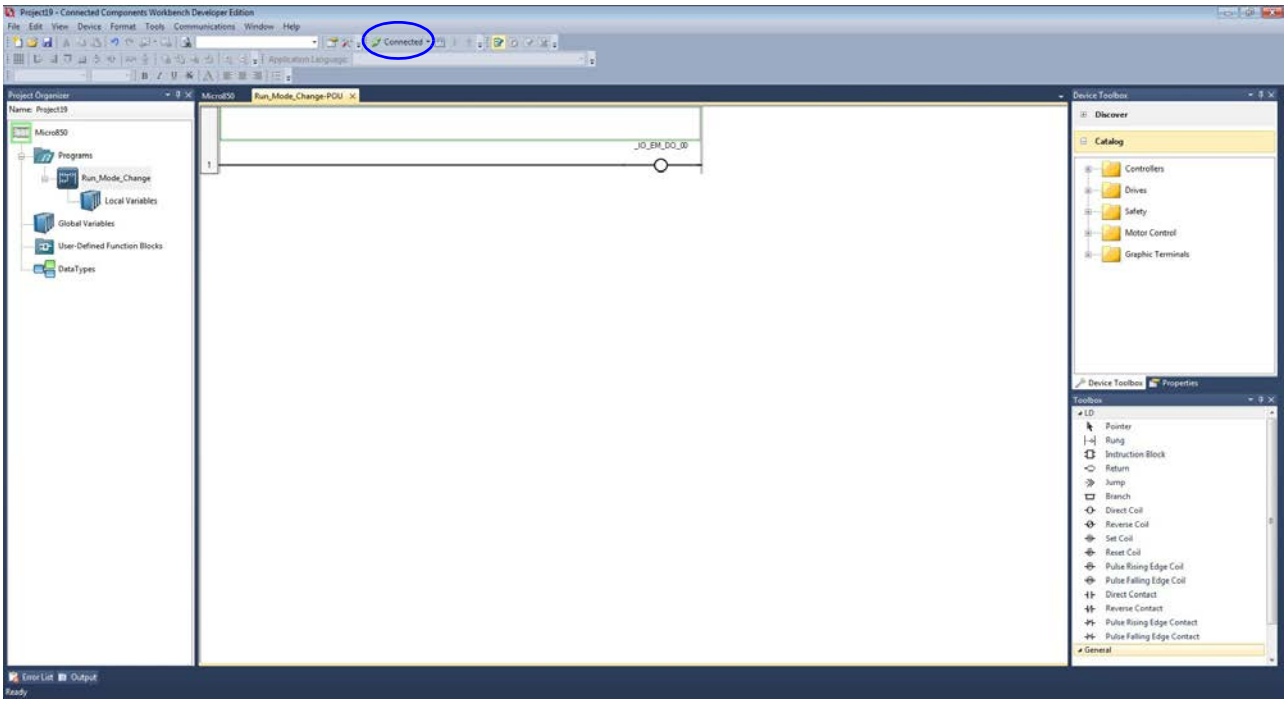
IMPORTANT From Connected Components Workbench version 8.0 onwards, selecting "Yes" to change the controller to Remote Run mode after a downloading a project automatically switches it to Debug mode.

Edit the Project Using Run Mode Change

Run Mode Change Toolbar



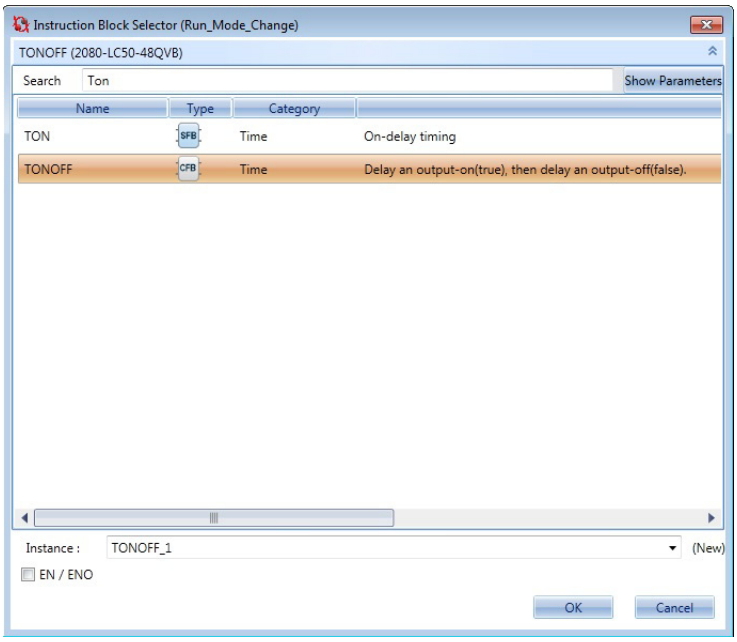
1. Click the Run Mode Change icon.
   Observe that the controller goes into Edit mode and is still connected.



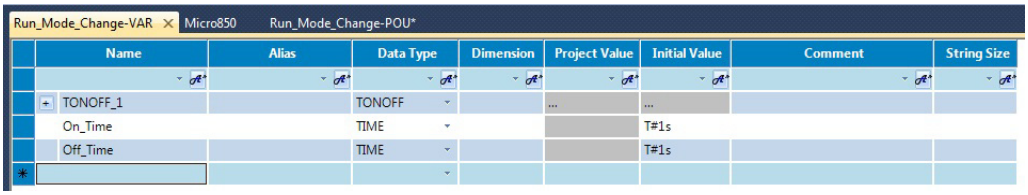
If you add a new variable during RMC, external data access and changing the access type (default is Read/Write) of this new variable is not available until you have chosen to Accept or Undo the Test Logic changes.

2. From the Toolbox, double-click Instruction Block to add it to the rung, or drag and drop Instruction Block onto the rung.

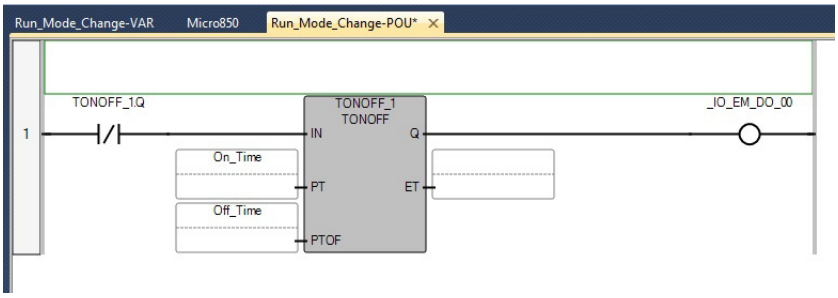
3. Double-click the newly added Instruction Block and select "Timer On/Off "(TONOFF).



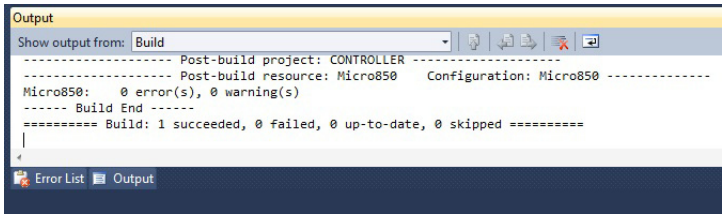
Configure the Instruction Block to trigger every one second.



4. From the Toolbox, double-click Reverse Contact to add it to the rung, or drag and drop Reverse Contact onto the run. Place it to left of the recently added Instruction Block.

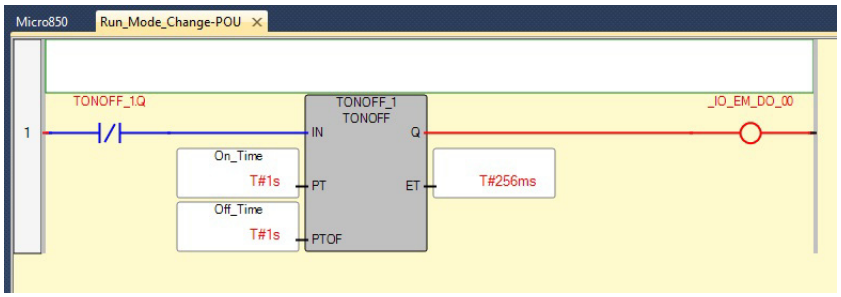


5. Click the Test Logic Changes icon to build the project and download it to the controller.


Output
Show output from: Build
---------- Post-build project: CONTROLLER ----------
---------- Post-build resource: Micro850 Configuration: Micro850 ----------
Micro850: 0 error(s), 0 warning(s)
------ Build End ------
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
Error List
Output

IMPORTANT When a Test Logic is performed, or undoing changes after the Test Logic is completed, any active communication instructions will be aborted while the changes are downloaded to the controller.

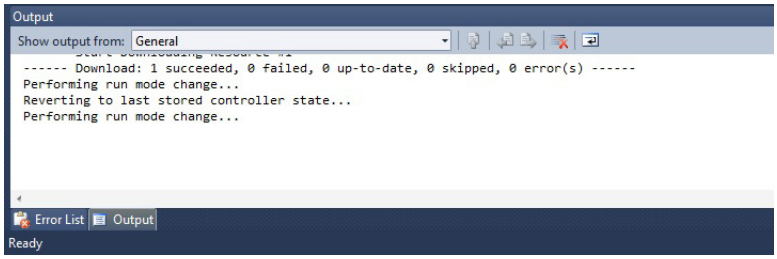
6. The controller will automatically go into Debug mode and display the updated project.



7. You can now choose to either Undo or Accept the changes to the project.

To Undo the Changes

1. Click the Undo Changes icon.
2. The changes will be discarded and the original project will be restored to the controller.


Output
Show output from: General
------ Download: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped, 0 error(s) ------
Performing run mode change...
Reverting to last stored controller state...
Performing run mode change...
Error List
Output
Ready

IMPORTANT When a Test Logic is performed, or undoing changes after the Test Logic is completed, any active communication instructions will be aborted while the changes are downloaded to the controller.

Observe that original project is shown and the controller is in Debug mode.

To Accept the Changes

1. Click the Accept Changes icon.
2. Observe that only the Run Mode Change icon is now enabled and the controller remains in Debug mode.

Notes:

User Interrupts

Interrupts allow you to interrupt your program based on defined events. This chapter contains information about using interrupts, the interrupt instructions, and interrupt configuration. The chapter covers the following topics:

Topic	Page
Information About Using Interrupts	237
User Interrupt Instructions	241
Using the Selectable Timed Interrupt (STI) Function	247
Selectable Time Interrupt (STI) Function Configuration and Status	247
Using the Event Input Interrupt (EII) Function	249

For more information on HSC Interrupt, see Use the High-Speed Counter and Programmable Limit Switch on page 127.

Information About Using Interrupts

The purpose of this section is to explain some fundamental properties of the User Interrupts, including:

- What is an interrupt?
- When can the controller operation be interrupted?
- Priority of User Interrupts
- Interrupt Configuration
- User Fault Routine

What is an Interrupt?

An interrupt is an event that causes the controller to suspend the Program Organization Unit (POU) it is currently performing, perform a different POU, and then return to the suspended POU at the point where it suspended. The Micro830 and Micro850 controllers support the following User Interrupts:

- User Fault Routine
- Event Interrupts (8)
- High-Speed Counter Interrupts (6)
- Selectable Timed Interrupts (4)
- Plug-in Module Interrupts (5)

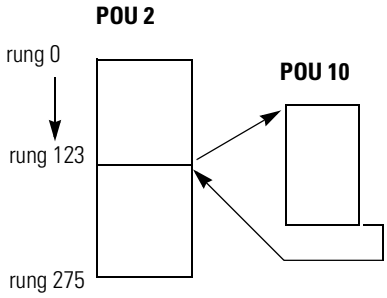
An interrupt must be configured and enabled to execute. When any one of the interrupts is configured (and enabled) and subsequently occurs, the user program:

1. suspends its execution of the current POU,
2. performs a predefined POU based upon which interrupt occurred, and
3. returns to the suspended operation.

Interrupt Operation Example

POU 2 is the main control program.
POU 10 is the interrupt routine.

- An Interrupt Event occurs at rung 123.
- POU 10 is executed.
- POU 2 execution resumes immediately after POU 10 is scanned.



Specifically, if the controller program is executing normally and an interrupt event occurs:

1. the controller stops its normal execution.
2. determines which interrupt occurred.
3. goes immediately to the beginning of the POU specified for that User Interrupt.
4. begins executing the User Interrupt POU (or set of POU/function blocks if the specified POU calls a subsequent function block).
5. completes the POU.
6. resumes normal execution from the point where the controller program was interrupted

When Can the Controller Operation be Interrupted?

The Micro830 controllers allow interrupts to be serviced at any point of a program scan. Use UID/ UIE instructions to protect program block which should not be interrupted.

Priority of User Interrupts

When multiple interrupts occur, the interrupts are serviced based upon their individual priority.

When an interrupt occurs and another interrupt(s) has already occurred but has not been serviced, the new interrupt is scheduled for execution based on its priority relative to the other pending interrupts. At the next point in time when an interrupt can be serviced, all the interrupts are executed in the sequence of highest priority to lowest priority.

If an interrupt occurs while a lower priority interrupt is being serviced (executed), the currently executing interrupt routine is suspended, and the higher priority interrupt is serviced. Then the lower priority interrupt is allowed to complete before returning to normal processing.

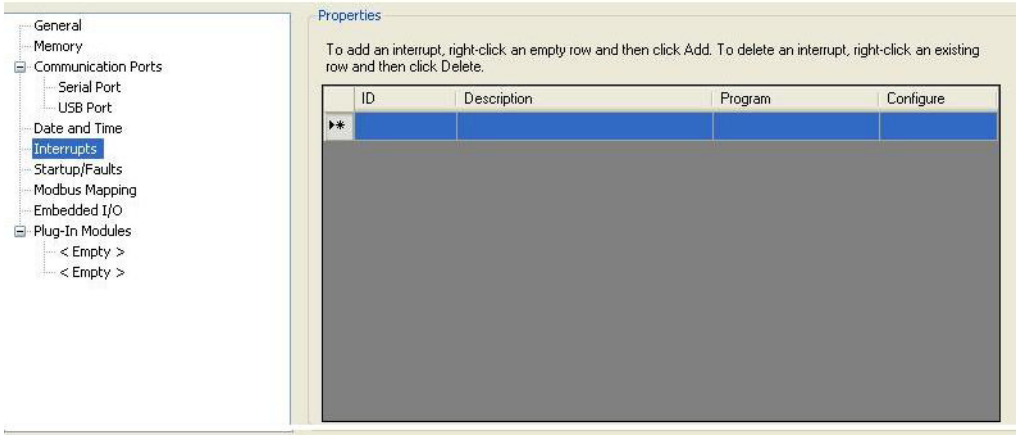
If an interrupt occurs while a higher priority interrupt is being serviced (executed), and the pending bit has been set for the lower priority interrupt, the currently executing interrupt routine continues to completion. Then the lower priority interrupt runs before returning to normal processing.

The priorities from highest to lowest are:

User Fault Routine	**highest priority**
Event Interrupt0	
Event Interrupt1	
Event Interrupt2	
Event Interrupt3	
High-Speed Counter Interrupt0	
High-Speed Counter Interrupt1	
High-Speed Counter Interrupt2	
High-Speed Counter Interrupt3	
High-Speed Counter Interrupt4	
High-Speed Counter Interrupt5	
Event Interrupt4	
Event Interrupt5	
Event Interrupt6	
Event Interrupt7	
Selectable Timed Interrupt0	
Selectable Timed Interrupt1	
Selectable Timed Interrupt2	
Selectable Timed Interrupt3	
Plug-In Module Interrupt0, 1, 2, 3, 4	**lowest priority**

User Interrupt Configuration

User interrupts can be configured and set as AutoStart from the Interrupts window.



User Fault Routine

The user fault routine gives you the option of doing the cleanup before a controller shutdown, when a specific user fault occurs. The fault routine is executed when any user fault occurs. The fault routine is not executed for non-user faults.

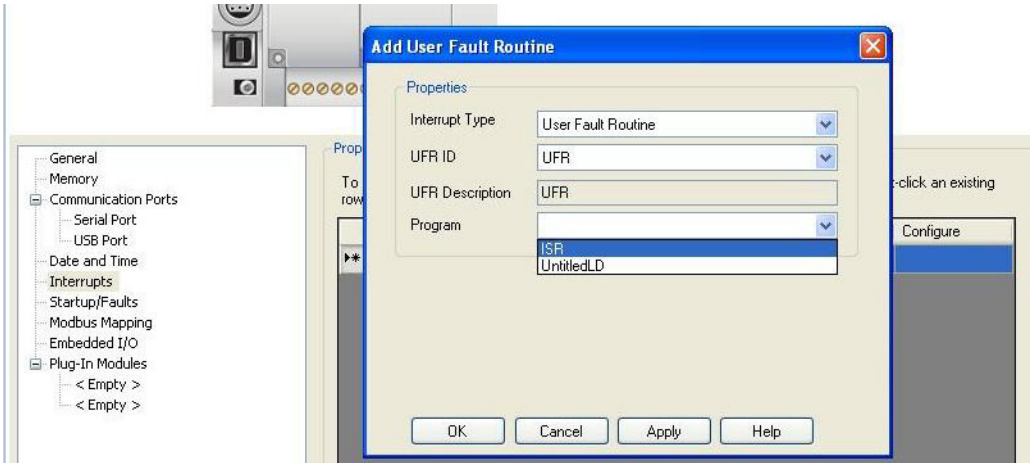
The controller goes to Fault mode after a User Fault Routine is executed, and the User Program execution stops.

Creating a User Fault Subroutine

To use the user fault subroutine:

1. Create a POU.

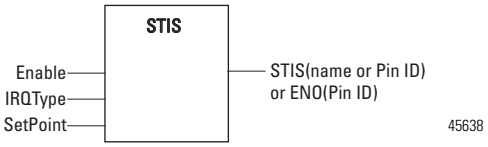
2. In the User Interrupt Configuration window, configure this POU as a User Fault routine.



User Interrupt Instructions

Instruction	Used To:	Page
STIS – Selectable Timed Start	Use the STIS (Selectable Timed Interrupt Start) instruction to the start the STI timer from the control program, rather than starting automatically.	241
UID – User Interrupt Disable	Use the User Interrupt Disable (UID) and the User Interrupt Enable (UIE) instructions to create zones in which user interrupts cannot occur.	242
UIE – User Interrupt Enable		244
UIF – User Interrupt Flush	Use the UIF instruction to remove selected pending interrupts from the system.	245
UIC – User Interrupt Clear	Use this function to clear Interrupt Lost bit for the selected User Interrupt(s).	246

STIS - Selectable Timed Start



STI0 is used in this document to define how STIS works.

STIS Parameters

Parameter	Parameter Type	Data Type	Parameter Description
Enable	Input	BOOL	Enable Function. When Enable = TRUE, function is performed. When Enable = FALSE, function is not performed.
IRQType	Input	UDINT	Use the STI defined DWORD IRQ_STI0, IRQ_STI1, IRQ_STI2, IRQ_STI3
SetPoint	Input	UINT	The user timer interrupt interval time value in milliseconds. When SetPoint = 0, STI is disabled. When SetPoint = 1…65535, STI is enabled.
STIS or ENO	Output	BOOL	Rung Status (same as Enable)

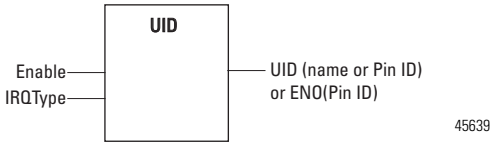
The STIS instruction can be used to start and stop the STI function or to change the time interval between STI user interrupts. The STI instruction has two operands:

- **IRQType** — This is the STI ID that a user wants to drive.
- **SetPoint** — This is the amount of time (in milliseconds) which must expire prior to executing the selectable timed user interrupt. A value of zero disables the STI function. The time range is from 0...65,535 milliseconds.

The STIS instruction applies the specified set point to the STI function as follows (STI0 is used here as an example):

- If a zero set point is specified, the STI is disabled and STI0.Enable is cleared (0).
- If the STI is disabled (not timing) and a value greater than 0 is entered into the set point, the STI starts timing to the new set point and STI0.Enable is set (1).
- If the STI is currently timing and the set point is changed, the new setting takes effect immediately, restarting from zero. The STI continues to time until it reaches the new set point.

UID - User Interrupt Disable



The UID instruction is used to disable selected user interrupts. The table below shows the types of interrupts with their corresponding disable bits:

Types of Interrupts Disabled by the UID Instruction

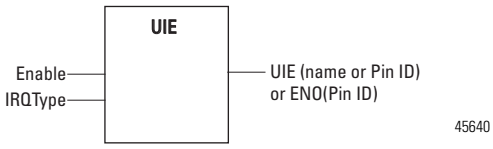
Interrupt Type	Element	Decimal Value	Corresponding Bit
Plug-In Module	UPM4	8388608	bit 23
Plug-In Module	UPM3	4194304	bit 22
Plug-In Module	UPM2	2097152	bit 21
Plug-In Module	UPM1	1048576	bit 20
Plug-In Module	UPM0	524288	bit 19
STI - Selectable Timed Interrupt	STI3	262144	bit 18
STI - Selectable Timed Interrupt	STI2	131072	bit 17
STI - Selectable Timed Interrupt	STI1	65536	bit 16
STI - Selectable Timed Interrupt	STI0	32768	bit 15
EII - Event Input Interrupt	Event 7	16384	bit 14
EII - Event Input Interrupt	Event 6	8192	bit 13
EII - Event Input Interrupt	Event 5	4096	bit 12
EII - Event Input Interrupt	Event 4	2048	bit 11
HSC - High-Speed Counter	HSC5	1024	bit 10
HSC - High-Speed Counter	HSC4	512	bit 9
HSC - High-Speed Counter	HSC3	256	bit 8
HSC - High-Speed Counter	HSC2	128	bit 7
HSC - High-Speed Counter	HSC1	64	bit 6
HSC - High-Speed Counter	HSC0	32	bit 5
EII - Event Input Interrupt	Event 3	16	bit 4
EII - Event Input Interrupt	Event 2	8	bit 3
EII - Event Input Interrupt	Event 1	4	bit 2
EII - Event Input Interrupt	Event 0	2	bit 1
UFR - User Fault Routine Interrupt	UFR	1	bit 0 (reserved)

To disable interrupt(s):

1. Select which interrupts you want to disable.
2. Find the Decimal Value for the interrupt(s) you selected.
3. Add the Decimal Values if you selected more than one type of interrupt.
4. Enter the sum into the UID instruction.

For example, to disable EII Event 1 and EII Event 3:
EII Event 1 = 4, EII Event 3 = 16
4 + 16 = 20 (enter this value)

UIE - User Interrupt Enable



The UIE instruction is used to enable selected user interrupts. The table below shows the types of interrupts with their corresponding enable bits:

Types of Interrupts Enabled by the UIE Instruction

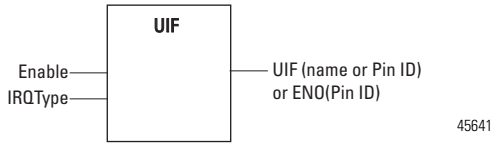
Interrupt Type	Element	Decimal Value	Corresponding Bit
Plug-In Module	UPM4	8388608	bit 23
Plug-In Module	UPM3	4194304	bit 22
Plug-In Module	UPM2	2097152	bit 21
Plug-In Module	UPM1	1048576	bit 20
Plug-In Module	UPM0	524288	bit 19
STI - Selectable Timed Interrupt	STI3	262144	bit 18
STI - Selectable Timed Interrupt	STI2	131072	bit 17
STI - Selectable Timed Interrupt	STI1	65536	bit 16
STI - Selectable Timed Interrupt	STI0	32768	bit 15
EII - Event Input Interrupt	Event 7	16384	bit 14
EII - Event Input Interrupt	Event 6	8192	bit 13
EII - Event Input Interrupt	Event 5	4096	bit 12
EII - Event Input Interrupt	Event 4	2048	bit 11
HSC - High-Speed Counter	HSC5	1024	bit 10
HSC - High-Speed Counter	HSC4	512	bit 9
HSC - High-Speed Counter	HSC3	256	bit 8
HSC - High-Speed Counter	HSC2	128	bit 7
HSC - High-Speed Counter	HSC1	64	bit 6
HSC - High-Speed Counter	HSC0	32	bit 5
EII - Event Input Interrupt	Event 3	16	bit 4
EII - Event Input Interrupt	Event 2	8	bit 3
EII - Event Input Interrupt	Event 1	4	bit 2
EII - Event Input Interrupt	Event 0	2	bit 1
		1	bit 0 (reserved)

To enable interrupt(s):

1. Select which interrupts you want to enable.
2. Find the Decimal Value for the interrupt(s) you selected.
3. Add the Decimal Values if you selected more than one type of interrupt.
4. Enter the sum into the UIE instruction.

For example, to enable EII Event 1 and EII Event 3:
EII Event 1 = 4, EII Event 3 = 16
4 + 16 = 20 (enter this value)

UIF - User Interrupt Flush



The UIF instruction is used to flush (remove pending interrupts from the system) selected user interrupts. The table below shows the types of interrupts with their corresponding flush bits:

Types of Interrupts Disabled by the UIF Instruction

Interrupt Type	Element	Decimal Value	Corresponding Bit
Plug-In Module	UPM4	8388608	bit 23
Plug-In Module	UPM3	4194304	bit 22
Plug-In Module	UPM2	2097152	bit 21
Plug-In Module	UPM1	1048576	bit 20
Plug-In Module	UPM0	524288	bit 19
STI - Selectable Timed Interrupt	STI3	262144	bit 18
STI - Selectable Timed Interrupt	STI2	131072	bit 17
STI - Selectable Timed Interrupt	STI1	65536	bit 16
STI - Selectable Timed Interrupt	STI0	32768	bit 15
EII - Event Input Interrupt	Event 7	16384	bit 14
EII - Event Input Interrupt	Event 6	8192	bit 13
EII - Event Input Interrupt	Event 5	4096	bit 12
EII - Event Input Interrupt	Event 4	2048	bit 11
HSC - High-Speed Counter	HSC5	1024	bit 10
HSC - High-Speed Counter	HSC4	512	bit 9
HSC - High-Speed Counter	HSC3	256	bit 8
HSC - High-Speed Counter	HSC2	128	bit 7
HSC - High-Speed Counter	HSC1	64	bit 6
HSC - High-Speed Counter	HSC0	32	bit 5
EII - Event Input Interrupt	Event 3	16	bit 4
EII - Event Input Interrupt	Event 2	8	bit 3
EII - Event Input Interrupt	Event 1	4	bit 2
EII - Event Input Interrupt	Event 0	2	bit 1
UFR - User Fault Routine Interrupt	UFR	1	bit 0 (reserved)

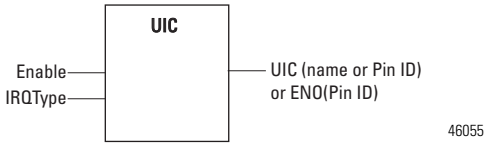
To flush interrupt(s):

1. Select which interrupts you want to flush.

2. Find the Decimal Value for the interrupt(s) you selected.
3. Add the Decimal Values if you selected more than one type of interrupt.
4. Enter the sum into the UIF instruction.

For example, to disable EII Event 1 and EII Event 3:
EII Event 1 = 4, EII Event 3 = 16
4 + 16 = 20 (enter this value)

UIC – User Interrupt Clear



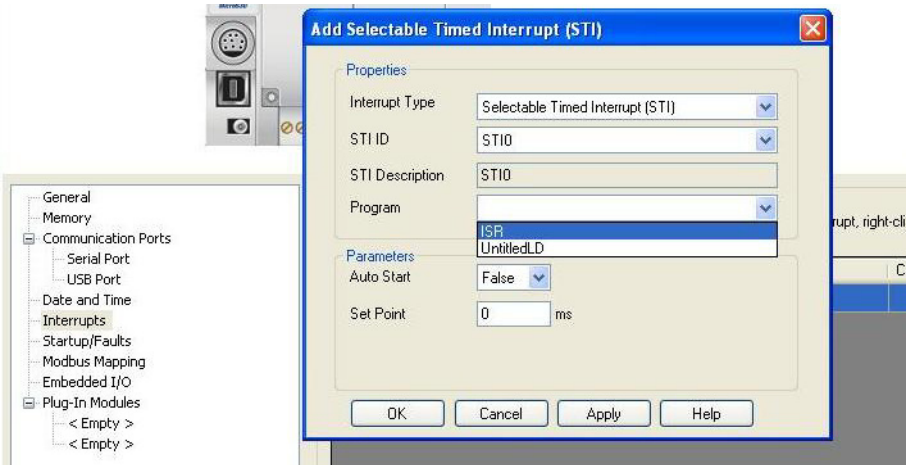
This C function clears Interrupt Lost bit for the selected User Interrupt(s).

Types of Interrupts Disabled by the UIC Instruction

Interrupt Type	Element	Decimal Value	Corresponding Bit
Plug-In Module	UPM4	8388608	bit 23
Plug-In Module	UPM3	4194304	bit 22
Plug-In Module	UPM2	2097152	bit 21
Plug-In Module	UPM1	1048576	bit 20
Plug-In Module	UPM0	524288	bit 19
STI - Selectable Timed Interrupt	STI3	262144	bit 18
STI - Selectable Timed Interrupt	STI2	131072	bit 17
STI - Selectable Timed Interrupt	STI1	65536	bit 16
STI - Selectable Timed Interrupt	STI0	32768	bit 15
EII - Event Input Interrupt	Event 7	16384	bit 14
EII - Event Input Interrupt	Event 6	8192	bit 13
EII - Event Input Interrupt	Event 5	4096	bit 12
EII - Event Input Interrupt	Event 4	2048	bit 11
HSC - High-Speed Counter	HSC5	1024	bit 10
HSC - High-Speed Counter	HSC4	512	bit 9
HSC - High-Speed Counter	HSC3	256	bit 8
HSC - High-Speed Counter	HSC2	128	bit 7
HSC - High-Speed Counter	HSC1	64	bit 6
HSC - High-Speed Counter	HSC0	32	bit 5
EII - Event Input Interrupt	Event 3	16	bit 4
EII - Event Input Interrupt	Event 2	8	bit 3
EII - Event Input Interrupt	Event 1	4	bit 2
EII - Event Input Interrupt	Event 0	2	bit 1
UFR - User Fault Routine Interrupt	UFR	1	bit 0 (reserved)

Using the Selectable Timed Interrupt (STI) Function

Configure the STI function from the Interrupt Configuration window.



The Selectable Timed Interrupt (STI) provides a mechanism to solve time critical control requirements. The STI is a trigger mechanism that allows you to scan or solve control program logic that is time sensitive.

Example of where you would use the STI are:

- PID type applications, where a calculation must be performed at a specific time interval.
- A block of logic that needs to be scanned more often.

How an STI is used is typically driven by the demands/requirements of the application. It operates using the following sequence:

1. The user selects a time interval.
2. When a valid interval is set and the STI is properly configured, the controller monitors the STI value.
3. When the time period has elapsed, the controller's normal operation is interrupted.
4. The controller then scans the logic in the STI POU.
5. When the STI POU is completed, the controller returns to where it was prior to the interrupt and continues normal operation.

Selectable Time Interrupt (STI) Function Configuration and Status

This section covers the configuration and status management of the STI function.

STI Function Configuration

STI Program POU

This is the name of the Program Organizational Unit (POU) which is executed immediately when this STI Interrupt occurs. You can choose any pre-programmed POU from the drop-down list.

STI Auto Start (STI0.AS)

Sub-Element Description	Data Format	User Program Access
AS - Auto Start	binary (bit)	read only

The AS (Auto Start) is a control bit that can be used in the control program. The auto start bit is configured with the programming device and stored as part of the user program. The auto start bit automatically sets the STI Timed Interrupt Enable (STI0.Enabled) bit when the controller enters any executing mode.

STI Set Point Milliseconds Between Interrupts (STI0.SP)

Sub-Element Description	Data Format	Range	User Program Access
SP - Set Point Msec	word (INT)	0…65,535	read/write

When the controller transitions to an executing mode, the SP (set point in milliseconds) value is loaded into the STI. If the STI is configured correctly, and enabled, the POU in the STI configuration is executed at this interval. This value can be changed from the control program by using the STIS instruction.

TIP The minimum value cannot be less than the time required to scan the STI POUplus the Interrupt Latency.

STI Function Status Information

STI Function status bits can be monitored either in the User Program, or in Connected Components Workbench, in Debug mode.

STI User Interrupt Executing (STI0.EX)

Sub-Element Description	Data Format	User Program Access
EX - User Interrupt Executing	binary (bit)	read only

The EX (User Interrupt Executing) bit is set whenever the STI mechanism completes timing and the controller is scanning the STI POU. The EX bit is cleared when the controller completes processing the STI subroutine.

The STI EX bit can be used in the control program as conditional logic to detect if an STI interrupt is executing.

STI User Interrupt Enable (STI0.Enabled)

Sub-Element Description	Data Format	User Program Access
Enabled - User Interrupt Enable	binary (bit)	read only

The User Interrupt Enable bit is used to indicate STI enable or disable status.

STI User Interrupt Lost (STI0.LS)

Sub-Element Description	Data Format	User Program Access
LS - User Interrupt Lost	binary (bit)	read/write

The LS is a status flag that indicates an interrupt was lost. The controller can process 1 active and maintain up to 1 pending user interrupt conditions before it sets the lost bit.

This bit is set by the controller. It is up to the control program to utilize, track, the lost condition if necessary.

STI User Interrupt Pending (STI0.PE)

Sub-Element Description	Data Format	User Program Access
PE - User Interrupt Pending	binary (bit)	read only

The PE is a status flag that represents an interrupt is pending. This status bit can be monitored or used for logic purposes in the control program if you need to determine when a subroutine cannot execute immediately.

This bit is automatically set and cleared by the controller. The controller can process 1 active and maintain up to 1 pending user interrupt conditions before it sets the lost bit.

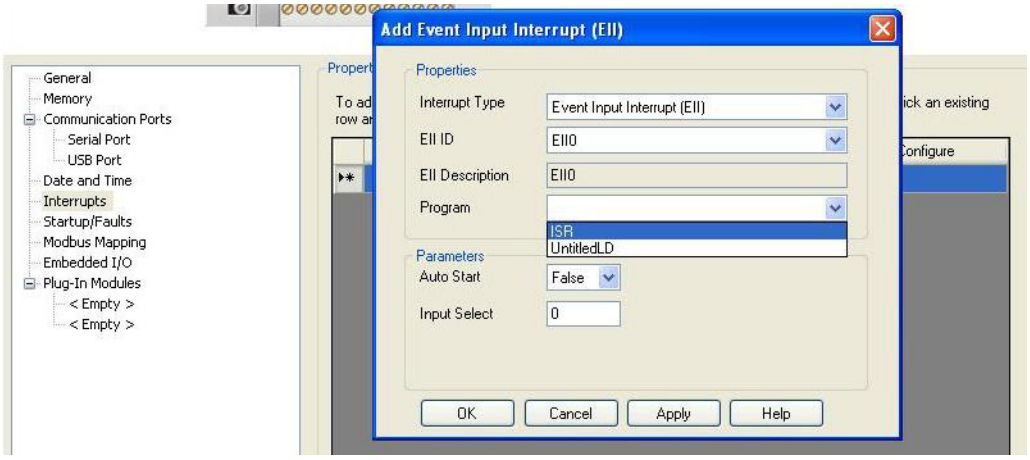
Using the Event Input Interrupt (EII) Function

The EII (Event Input Interrupt) is a feature that allows the user to scan a specific POU when an input condition is detected from a field device.

EII0 is used in this document to define how EII works.

Configure EII Input Edge from the Embedded I/O configuration window.

Configure the EII from the Interrupt Configuration window.



Event Input Interrupt (EII) Function Configuration and Status

EII Function Configuration

The Event Input Interrupt Function has the following related configuration parameters.

EII Program POU

This is the name of the Program Organizational Unit (POU) which is executed immediately when this EII Interrupt occurs. You can choose any pre-programmed POU from the drop-down list.

EII Auto Start (EII0.AS)

Sub-Element Description	Data Format	User Program Access
AS - Auto Start	binary (bit)	read only

AS (Auto Start) is a control bit that can be used in the control program. The auto start bit is configured with the programming device and stored as part of the user program. The auto start bit automatically sets the Event User Interrupt Enable bit when the controller enters any executing mode.

EII Input Select (EII0.IS)

Sub-Element Description	Data Format	User Program Access
IS - Input Select	word (INT)	read only

The IS (Input Select) parameter is used to configure each EII to a specific input on the controller. Valid inputs are 0...N, where N is either 15, or the maximum input ID, whichever is smaller.

This parameter is configured with the programming device and cannot be changed from the control program.

EII Function Status Information

EII Function status bits can be monitored either in the User Program, or in Connected Components Workbench, in Debug mode.

EII User Interrupt Executing (EII0.EX)

Sub-Element Description	Data Format	User Program Access
EX - User Interrupt Executing	binary (bit)	read only

The EX (User Interrupt Executing) bit is set whenever the EII mechanism detects a valid input and the controller is scanning the EII POU. The EII mechanism clears the EX bit when the controller completes its processing of the EII subroutine.

The EII EX bit can be used in the control program as conditional logic to detect if an EII interrupt is executing.

EII User Interrupt Enable (EII0.Enabled)

Sub-Element Description	Data Format	User Program Access
Enabled - User Interrupt Enable	binary (bit)	read only

The Enabled (User Interrupt Enable) bit is used to indicate the EII enable or disable status.

EII User Interrupt Lost (EII0.LS)

Sub-Element Description	Data Format	User Program Access
LS - User Interrupt Lost	binary (bit)	read/write

LS (User Interrupt Lost) is a status flag that represents an interrupt has been lost. The controller can process 1 active and maintain up to 1 pending user interrupt conditions before it sets the lost bit.

This bit is set by the controller. It is up to the control program to utilize or track, the lost condition if necessary.

EII User Interrupt Pending (EII0.PE)

Sub-Element Description	Data Format	User Program Access
PE - User Interrupt Pending	binary (bit)	read only

PE (User Interrupt Pending) is a status flag that represents an interrupt is pending. This status bit can be monitored, or used for logic purposes, in the

control program if you need to determine when a subroutine cannot execute immediately.

This bit is automatically set and cleared by the controller. The controller can process 1 active and maintain up to 1 pending user interrupt conditions before it sets the lost bit.

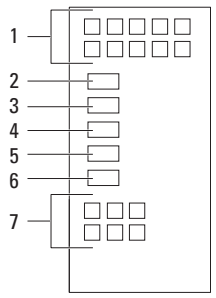
Troubleshooting

Status Indicators on the Controller

Micro830 Controllers

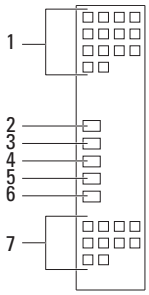
Status Indicators

10/16 Point Controllers



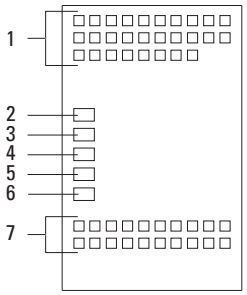
45031a

24 Point Controllers



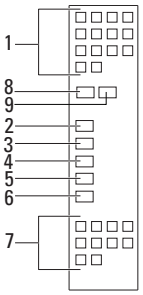
45017a

48 Point Controllers

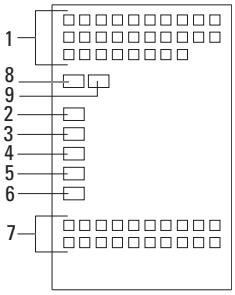


45037a

Micro850 Controllers



45934



45935

Status Indicator Description

	Description	State	Indicates
1	Input status	Off	Input is not energized
		On	Input is energized (terminal status)
2	Power status	Off	No input power, or power error condition
		Green	Power on
3	Run status	Off	Not executing the user program
		Green	Executing the user program in run mode
		Flashing green	Memory module transfer in progress

Status Indicator Description

	Description	State	Indicates
4	Fault status	Off	No fault detected.
		Red	Non-recoverable fault that requires a power cycle.
		Flashing Red	Recoverable fault.
5	Force status	Off	No force conditions are active.
		Amber	Force conditions are active.
6	Serial communications status	Off	No traffic for RS-232/RS-485.
		Green	Traffic through RS-232/RS-485. The indicator only blinks when transmitting data. It does not blink when receiving data.
7	Output status	Off	Output is not energized.
		On	Output is energized (logic status).
8	Module status	Steady Off	No power.
		Flashing Green	Standby.
		Steady Green	Device operational.
		Flashing Red	Minor fault (minor and major recoverable faults).
		Steady Red	Major Fault (non-recoverable fault).
		Flashing Green and Red	Self-test.
9	Network status	Steady Off	Not powered, no IP address. The device is powered off, or is powered on but with no IP address.
		Flashing Green	No connections. An IP address is configured, but no Ethernet application is connected.
		Steady Green	Connected. At least one EtherNet/IP session is established.
		Flashing Red	Connection timeout (not implemented).
		Steady Red	Duplicate IP. The device has detected that its IP address is being used by another device in the network. This status is applicable only if the device's duplicate IP address detection (ACD) feature is enabled.
		Flashing Green and Red	Self-test. The device is performing power-on self-test (POST). During POST, the network status indicator alternates flashing green and red.

Normal Operation

The POWER and RUN indicators are on. If a force condition is active, the FORCE indicator turns on and remains on until all forces are removed.

Error codes

This section lists possible error codes for your controller, as well as recommended actions for recovery. Information about the fault is stored in a fault log, which can be accessed from the Diagnostics page in Connected Components Workbench software. The fault log contains brief information about the last fault, and detailed information about the last 10 non-recoverable faults that occurred.

If an error persists after performing the recommended action, contact your local Rockwell Automation technical support representative. For contact information, go to
http://support.rockwellautomation.com/MySupport.asp

Fault Types

There are two basic types of faults that can occur:

- Recoverable – A recoverable fault can be cleared without having to power cycle the controller. The fault LED flashes red when a recoverable fault occurs.
- Non-recoverable – A non-recoverable fault requires the controller to be power cycled before clearing the fault. Non-recoverable controller faults may become recoverable after the controller has been power cycled, or an automatic reset is performed by the controller. If the controller performs an automatic reset and the fault becomes recoverable, the fault will not be logged. After the controller has been power cycled or reset, check the fault log in the Diagnostic page of the Connected Components Workbench software, then clear the fault. The fault LED is solid red when a non-recoverable fault occurs.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xF000	Recoverable	The controller was unexpectedly reset due to a noisy environment or an internal hardware failure. If the system variable _SYSVA_USER_DATA_LOST is set, the controller is able to recover the user program but the user data is cleared. If not, the Micro800 controller program is cleared.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Check wiring to eliminate any noise, refer to Wiring Requirements and Recommendation on page 39.
0xF001	Recoverable	The controller program has been cleared. This happened because: • a power-down occurred during program download or data transfer from the memory module • the cable was removed from the controller during program download. • the RAM integrity test failed.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Transfer the program using the memory module restore utility.
0xF002	Non-recoverable	The controller hardware watchdog was activated. The controller hardware watchdog timeout happens if program scan is more than three seconds. If the system variable _SYSVA_USER_DATA_LOST is set, the controller is able to recover the user program but the user data is cleared. If not, the Micro800 controller program is cleared.	See Corrective Actions for Non-recoverable Faults on page 263.
0xF003	Recoverable	One of the following occurred: • The memory module hardware faulted. • The memory module connection faulted. • The memory module was incompatible with the Micro800 controller's firmware revision.	Perform one of the following: • Remove the memory module and plug it in again. • Obtain a new memory module. • See Corrective Actions for Recoverable Faults on page 263. • Upgrade the Micro800 controller's firmware revision to be compatible with the memory module. For more information on firmware revision compatibility, go to http://www.rockwellautomation.com/support/firmware.html
0xF004	Recoverable	A failure occurred during the memory module data transfer.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Attempt the data transfer again. If the error persists, replace the memory module. • For Embedded RTC failure, restart the controller.
0xF005	Recoverable	The user program failed an integrity check while the Micro800 controller was in Run mode.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Refer to the Wire Your Controller on page 39.
0xF006	Recoverable	The user program is incompatible with the Micro800 controller's firmware revision.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Contact your local Rockwell Automation technical support representative.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xF010	Recoverable	The user program contains a function/function block that is not supported by the Micro800 controller.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Contact your local Rockwell Automation technical support representative.
0xF014	Recoverable	A memory module memory error occurred.	• See Corrective Actions for Recoverable Faults on page 263. • Reprogram the memory module. If the error persists, replace the memory module.
0xF015	Non-recoverable	An unexpected software error occurred.	Perform one of the following: • See Corrective Actions for Non-recoverable Faults on page 263. • Refer to Wiring Requirements and Recommendation on page 39.
0xF016	Non-recoverable	An unexpected hardware error occurred.	Perform one of the following: • See Corrective Actions for Non-recoverable Faults on page 263. • Refer to Wiring Requirements and Recommendation on page 39.
0xF017	Non-recoverable	An unexpected software error occurred due to unexpected hardware interrupt. If the system variable _SYSVA_USER_DATA_LOST is set, the controller is able to recover the user program but the user data is cleared. If not, the Micro800 controller program is cleared.	Perform one of the following: • See Corrective Actions for Non-recoverable Faults on page 263. • Refer to Wiring Requirements and Recommendation on page 39.
0xF018	Non-recoverable	An unexpected software error occurred due to SPI communication failure. If the system variable _SYSVA_USER_DATA_LOST is set, the controller is able to recover the user program but the user data is cleared. If not, the Micro800 controller program is cleared.	Perform one of the following: • See Corrective Actions for Non-recoverable Faults on page 263. • Refer to Wiring Requirements and Recommendation on page 39.
0xF019	Non-recoverable	An unexpected software error occurred due to memory or other controller resource issue.	See Corrective Actions for Non-recoverable Faults on page 263.
0xF01A	Recoverable	The controller was unexpectedly reset during Run Mode Change (RMC) due to a noisy environment or an internal hardware failure. If the system variable _SYSVA_USER_DATA_LOST is set, the controller is able to recover the user program but the user data is cleared. If not, the Micro800 controller program is cleared.	See Corrective Actions for Recoverable Faults on page 263.
0xF020	Recoverable	The base hardware faulted or is incompatible with the Micro800 controller's firmware revision.	See Corrective Actions for Recoverable Faults on page 263.
0xF021	Recoverable	The I/O configuration in the user program is invalid or does not exist in the Micro800 controller.	See Corrective Actions for Recoverable Faults on page 263.
0xF022	Recoverable	The user program in the memory module is incompatible with the Micro800 controller's firmware revision.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Replace the memory module.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xF023	Non-recoverable	The controller program has been cleared. This happened because: • a power down occurred during program download or transfer from the memory module. • the Flash Integrity Test failed (Micro810 only).	Perform one of the following: • See Corrective Actions for Non-recoverable Faults on page 263. • Download or transfer the program.
0xF030 0xF031 0xF032 0xF033	Recoverable	Power down information in persistent memory may not be written properly due to a noisy environment or an internal hardware failure. If the system variable _SYSVA_USER_DATA_LOST is set, the controller is able to recover the user program but the user data is cleared. If not, the Micro800 controller program is cleared.	See Corrective Actions for Recoverable Faults on page 263.
0xF050	Recoverable	The embedded I/O configuration in the user program is invalid.	See Corrective Actions for Recoverable Faults on page 263.
0xF100	Recoverable	There is general configuration error detected in the motion configuration downloaded from the Connected Components Workbench software, such as number of axis, or motion execution interval being configured out of range.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Correct the axes configuration in the user program.
0xF110	Recoverable	There is motion resource missing, such as Motion_DIAG variable not defined.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Correct the axes configuration in the user program.
0xF12z[1]	Recoverable	Motion configuration for axis z cannot be supported by this controller model, or the axis configuration has some resource conflict with some other motion axis, which has been configured earlier.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Remove all axes and re-configure motion with the guidance from the User Manual.
0xF15z[1]	Recoverable	There is a motion engine logic error (firmware logic issue or memory crash) for one axis detected during motion engine cyclic operation. One possible reason can be motion engine data/ memory crash.	See Corrective Actions for Recoverable Faults on page 263.
0xF210	Recoverable	The expansion I/O terminator is missing.	Perform the following: 1. Power off the controller. 2. Attach the expansion I/O terminator on the last expansion I/O module on the system. 3. Power on the controller. 4. See Corrective Actions for Recoverable Faults on page 263.
0xF230	Recoverable	The maximum number of expansion I/O modules has been exceeded.	Perform the following: 1. Power off the controller. 2. Check that the number of expansion I/O modules is not more than four. 3. Power on the controller. 4. See Corrective Actions for Recoverable Faults on page 263.
0xF250	Recoverable	There is a non-recoverable error and the expansion I/O module(s) could not be detected.	See Corrective Actions for Recoverable Faults on page 263.
0xF26z[2]	Recoverable	An expansion I/O master fault is detected on the system.	See Corrective Actions for Recoverable Faults on page 263.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xF27z[2]	Recoverable	A non-recoverable communication fault has occurred on the expansion I/O module.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Replace the slot number z module.
0xF28z[2]	Recoverable	Expansion I/O baudrate error.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Replace the slot number z module.
0xF29z[2]	Recoverable	A module fault is detected on your expansion I/O module.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Replace the slot number z module.
0xF2Az[2]	Recoverable	Expansion I/O power failure	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Replace the slot number z module.
0xF2Bz[2]	Recoverable	Expansion I/O configuration fault.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Correct the expansion IO module configuration in the user program to match that of the actual hardware configuration. • Check the expansion I/O module operation and condition. • Replace the expansion I/O module.
0xF300	Recoverable	The memory module is present but memory module is empty and restore operation is requested.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Check to make sure there is a valid project in the memory module. • Download a user program and use the backup function to the memory module.
0xF301	Recoverable	The memory module's project is not compatible with the controller.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Check to make sure there is a user program with a controller that has the correct controller catalog configured. • Download a user program and use the backup function to the memory module.
0xF302	Recoverable	The password is mismatched between memory module and controller. Only applies to Micro820 controller when Remote LCD performs the restore operation. This fault does not apply to Micro800 controller firmware revision 10 and later.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Check to make sure that the user program in the memory module has the correct password. • Download a user program with a password and use the backup function to the memory module. • Use Connected Components Workbench to enter the correct password into the controller and perform the restore operation again.
0xF303	Recoverable	The memory module is not present and restore operation is requested.	Check to make sure the memory module is present.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xF0Az[3]	Recoverable	The plug-in I/O module experienced an error during operation.	Perform the following: • Check the condition and operation of the plug-in I/O module. • See Corrective Actions for Recoverable Faults on page 263.
0xF0Bz[3]	Recoverable	The plug-in I/O module configuration does not match the actual I/O configuration detected.	Perform one of the following: • See Corrective Actions for Recoverable Faults on page 263. • Correct the plug-in I/O module configuration in the user program to match that of the actual hardware configuration. • Check the condition and operation of the plug-in I/O module. • Replace the plug-in I/O module.
0xF0Dz[3]	Recoverable	When power was applied to the plug-in I/O module or the plug-in I/O module was removed, a hardware error occurred.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the plug-in I/O module configuration in the user program. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.
0xF0Ez[3]	Recoverable	The plug-in I/O module configuration does not match the actual I/O configuration detected.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the plug-in I/O module configuration in the user program. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.
0xF830	Recoverable	An error occurred in the EII configuration.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Review and change the EII configuration in the Micro800 controller properties.
0xF840	Recoverable	An error occurred in the HSC configuration.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Review and change the EII configuration in the Micro800 controller properties.
0xF850	Recoverable	An error occurred in the STI configuration.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Review and change the EII configuration in the Micro800 controller properties.
0xF860	Recoverable	A data overflow occurred. A data overflow error is generated when the ladder, structured text or function block diagram execution encounters a divide-by-zero.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the program to ensure that there is no data overflow. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.
0xF870	Recoverable	An index address was out of data space.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the program to ensure that there is no index used to access an array element beyond the array boundaries. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xF0878	Recoverable	An index used to access a bit is beyond the boundaries of the data type it is used on.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the program to ensure that there is no index used to access a bit beyond the boundaries of the data type. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.
0xF880	Recoverable	A data conversion error occurred.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the program to ensure that there is no data conversion error. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.
0xF888	Recoverable	The call stack of the controller cannot support the sequence of calls to function blocks in the current project. Too many blocks are within another block.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Change the project to reduce the quantity of blocks being called within a block.
0xF898	Recoverable	An error occurred in the user interrupt configuration for the plug-in I/O module.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Correct the user interrupt configuration for plug-in I/O module in the user program to match that of the actual hardware configuration.
0xF8A0	Recoverable	The TOW parameters are invalid.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the program to ensure that there are no invalid parameters. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.
0xF8A1	Recoverable	The DOY parameters are invalid.	Perform the following: 1. See Corrective Actions for Recoverable Faults on page 263. 2. Correct the program to ensure that there are no invalid parameters. 3. Build and download the program using Connected Components Workbench. 4. Put the Micro800 controller into Run mode.

List of Error Codes for Micro800 controllers

Error Code	Fault Type	Description	Recommended Action
0xFFzz[4]	Recoverable	A user-created fault from Connected Components Workbench has occurred.	See Corrective Actions for Recoverable Faults on page 263.
0xD00F	Recoverable	A particular hardware type (for example, embedded I/O) was selected in the user program configuration, but did not match the actual hardware base.	See Corrective Actions for Recoverable Faults on page 263.
0xD011	Recoverable	The program scan time exceeded the watchdog timeout value.	Perform the following: • See Corrective Actions for Recoverable Faults on page 263. • Determine if the program is caught in a loop and correct the problem. Fault may occur if your Structured Text program contains a For loop with the upper limit set to the maximum value of the variable. For example, the variable is a USINT and the limit is set to 255, or the variable is a UINT and the limit is set to 65535. To correct the fault, perform the following: 1. Correct the program to ensure that the upper limit is not reached. One method is to use a data type with a larger maximum value. 2. Build and download the program using Connected Components Workbench. 3. Put the Micro800 controller into Run mode. If your program is designed to have a scan time of longer than three seconds, in the user program, increase the watchdog timeout value that is set in the system variable _SYSVA_TCYWDG and then build and download the program using Connected Components Workbench.

(1) z indicates the logic axis ID. (0...3)

(2) z indicates the slot number of the expansion I/O. If z=0, then the slot number cannot be identified.

(3) z is the slot number of the plug-in module. If z = 0, then the slot number cannot be identified.

(4) zz indicates the last byte of the program number. Only program numbers up to 0xFF can be displayed. For program numbers 01x00 to 0xFFFF, only the last byte is displayed.)

Corrective Action for Recoverable and Non-recoverable Faults

Corrective Actions for Recoverable Faults

Perform the following:

1. Optionally save the fault log from Connected Components Workbench software.
2. Clear the recoverable fault using Connected Components Workbench software.
3. If problem persists, contact technical support with the fault log.

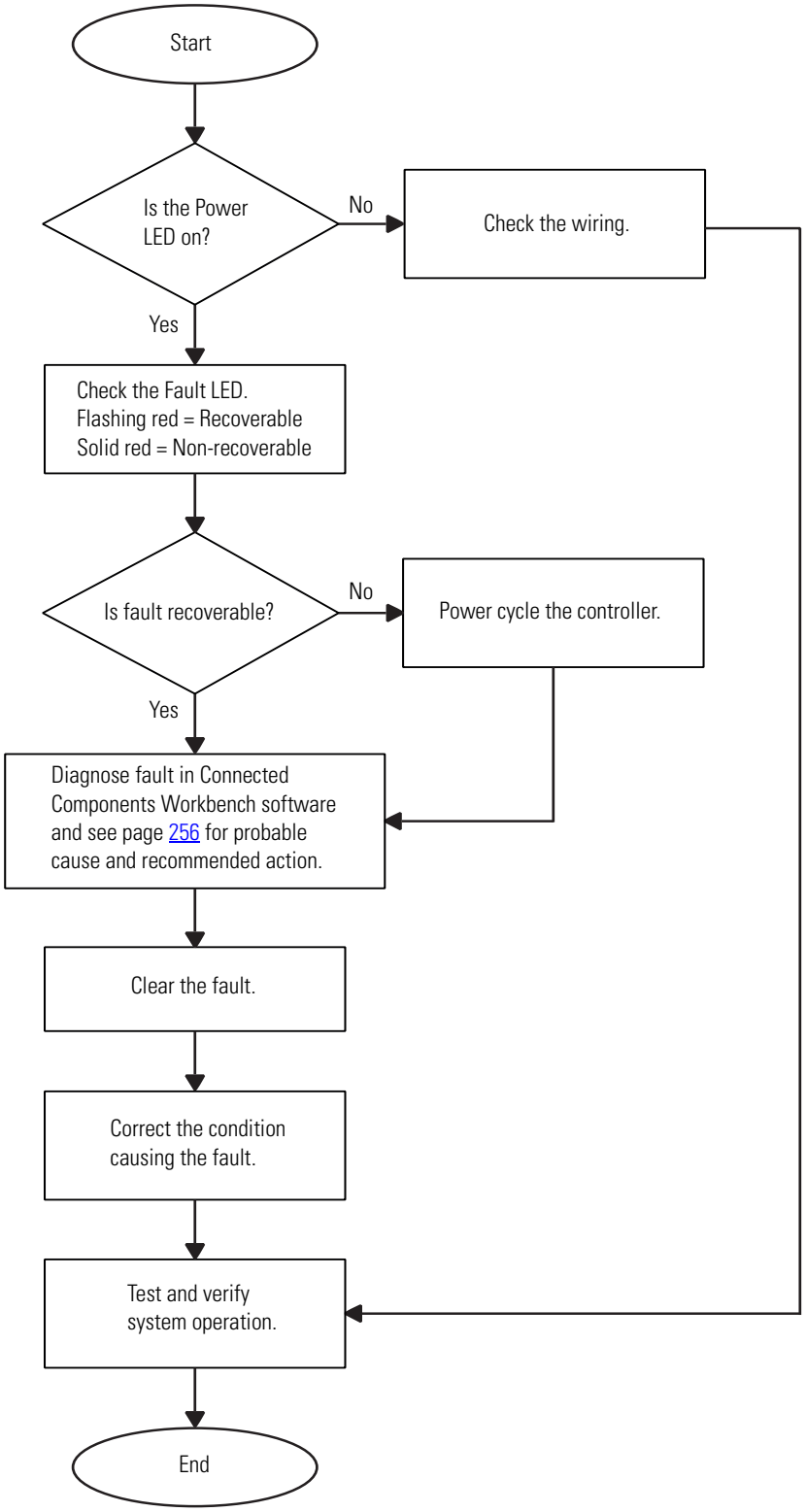
Corrective Actions for Non-recoverable Faults

Perform the following:

1. Power cycle your Micro800 controller.
2. Controller will go to recoverable fault. Optionally save the fault log from Connected Components Workbench software.
3. Clear the recoverable fault using Connected Components Workbench software.
4. If program is lost, build and download your program using Connected Components Workbench software.
5. If problem persists, contact technical support with the fault log.

Controller Error Recovery Model

Use the following error recovery model to help you diagnose software and hardware problems in the micro controller. The model provides common questions you might ask to help troubleshoot your system. Refer to the recommended pages within the model for further help.



Calling Rockwell Automation for Assistance

If you need to contact Rockwell Automation or local distributor for assistance, it is helpful to obtain the following (prior to calling):

- controller type, series letter, revision letter, and firmware (FRN) number of the controller
- controller indicator status

Notes:

PID Function Blocks

The PID function block has parameter naming similar to RSLogix 500 and is recommended for users who are already familiar with programming in RSLogix 500. The IPIDCONTROLLER function block has the advantage of supporting auto tune.

Comparison Between IPIDCONTROLLER and PID

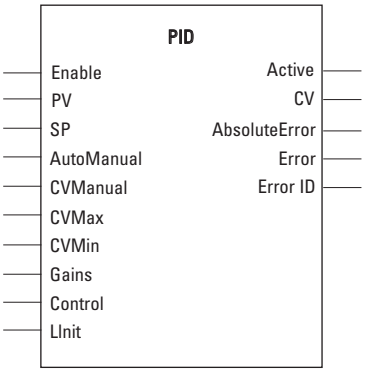
IPIDCONTROLLER	PID	Description
Common parameters		
Process	PV	Process Variable feedback
Setpoint	SP	Set Point input
Output	CV	CV output
Gains.DirectActing	Control	Control direction of process (cooling versus heating).
Gains.ProportionalGain	Gains.Kc	Controller gain for both P and I
Gains.TimeIntegral	Gains.Ti	Time integral value for I
Gains.TimeDerivative	Gains.Td	Time derivative value for D
Gains.DerivativeGain	Gains.FC	A higher filter constant makes CV output more responsive to error. Acts like a derivative gain.
AbsoluteError	AbsoluteError	Absolute value of error
PID specific parameters		
–	CVMin	For limiting CV
	CVMax	For limiting CV
	AutoManual	TRUE = Normal operation of PID. FALSE = Manual operation using CVManual.
	CVManual	CV when in manual mode
	Enable	TRUE = Start execution with current input parameters. FALSE = CV equals zero.
	Active	TRUE = PID state is running. FALSE = PID state is stopped.
	Error	TRUE = PID has an error. FALSE = PID has no errors.
	ErrorID	PID Error ID
IPIDCONTROLLER specific parameters		

Comparison Between IPIDCONTROLLER and PID

IPIDCONTROLLER	PID	Description
Auto	–	TRUE = Normal operation of PID. FALSE = Output tracks Feedback.
Feedback		Feedback of the control being applied to the process. Usually it's the PID's CV after any limits or manual control has been applied.
AutoTune		TRUE = Autotune. FALSE = No Autotune.
ATParameters		Autotune parameters
ATWarning		Autotune warning
OutGains		Gains from Autotune
Initialize		Used for AutoTune

PID Function Block

This function block diagram shows the arguments in the PID function block.



The following table explains the arguments used in this function block.

PID Arguments

Parameter	Parameter Type	Data Type	Description
Enable	Input	BOOL	Enable instruction. TRUE = Start execution with current input parameters. FALSE = CV equals zero.
PV	Input	REAL	Process Value. This value is typically read from an analog input module. The SI unit must be the same as Setpoint.
SP	Input	REAL	The set point value for the process.
AutoManual	Input	BOOL	Auto or manual mode selection: TRUE = Normal operation of PID. FALSE = Manual operation using CVManual.
CVManual	Input	REAL	Control value input defined for manual mode operation. The valid range for CVManual is: CVMin < CVManual < CVMax

PID Arguments

Parameter	Parameter Type	Data Type	Description
CVMin	Input	REAL	Control value minimum limit. If CV < CVMin, then CV = CVMin. If CVMin > CVMax, and error occurs.
CVMax	Input	REAL	Control value maximum limit. If CV > CVMax, then CV = CVMax. If CVMax < CVMin, an error occurs.
Gains	Input	PID_GAINS	Gains of PID for controller. Use the PID_GAINS data type to configure the Gains parameter.
Control	Input	BOOL	Control direction of the process: TRUE = Direct acting, such as Cooling. FALSE = Reverse acting, such as Heating.
LInit	Input	BOOL	Reserved for future use.
Active	Output	BOOL	Status of the PID controller: TRUE = PID state is running. FALSE = PID state is stopped.
CV	Output	REAL	The control value output. If any error occurred, CV is 0.
AbsoluteError	Output	REAL	Absolute error is the difference between process value (PV) and setpoint (SV) value.
Error	Output	BOOL	Indicates the existence of an error condition. TRUE = PID has an error. FALSE = PID has no errors.
ErrorID	Output	USINT	A unique numeric that identifies the error. The errors are defined in PID error codes.

GAIN_PID Data Type

Parameter	Parameter Type	Data Type	Description
Kc	Input	REAL	Controller gain for PID. Proportional and Integral are dependent on this gain (>= 0.0001). Increasing Kc improves response time but also increases overshoot and oscillation of the PID. If Kc is invalid, an error occurs.

GAIN_PID Data Type

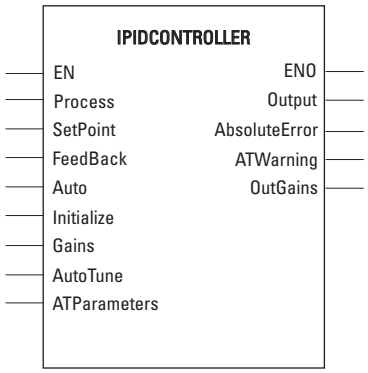
Parameter	Parameter Type	Data Type	Description
Ti	Input	REAL	Time integral constant in seconds (>= 0.0001). Increasing Ti decreases overshoot and oscillation of the PID. If Ti is invalid, an error occurs.
Td	Input	REAL	Time derivative constant in seconds (>= 0.0). When Td equals 0, then there is no derivative action and PID becomes a PI controller. Increasing Td reduces the overshot and removes the oscillation of the PID controller. If Td is invalid, an error occurs.
FC	Input	REAL	Filter constant (>= 0.0). Recommended range for FC is 0...20. Increasing FC smooths the response of the PID controller. If FC is invalid, an error occurs.

PID Error Codes

Error Code	Description
0	PID is working normally.
1	Kc is invalid.
2	Ti is invalid.
3	Td is invalid.
4	FC is invalid.
5	CVMin > CVMax, or CVMax < CVMin
6	CVManual < CVMin CVManaul is invalid.
7	CVManual > CVMax CVManual is invalid.

IPIDCONTROLLER Function Block

This function block diagram shows the arguments in the IPIDCONTROLLER function block.



The following table explains the arguments used in this function block.

IPIDCONTROLLER Arguments

Parameter	Parameter Type	Data Type	Description
EN	Input	BOOL	Function block enable TRUE = Execute function. FALSE = Do not execute function. Applicable to Ladder Diagram programs.
Process	Input	REAL	Process value, which is the value measured from the process output.
SetPoint	Input	REAL	The set point value for the process.
Feedback	Input	REAL	Feedback signal, which is the value of the control variable applied to the process. For example, the feedback can be IPIDCONTROLLER output.
Auto	Input	BOOL	Operating modes of PID controller: TRUE = Normal operation of PID. FALSE = Output tracks Feedback.
Initialize	Input	BOOL	A change in value (TRUE to FALSE or FALSE to TRUE) causes the controller to eliminate any proportional gain during that cycle. It also initializes AutoTune sequences.
Gains	Input	GAIN_PID	Gains PID for IPIDCONTROLLER. Use the GAIN_PID data type to define the parameters for the Gains input.
AutoTune	Input	BOOL	TRUE = Autotune. FALSE = No Autotune.
ATParameters	Input	AT_Param	AutoTune parameters Use AT_Param data type to define the parameters for the ATParameters input.
Output	Output	Real	Output value from the controller.
AbsoluteError	Output	Real	Absolute error (Process - SetPoint) from the controller.

IPIDCONTROLLER Arguments

Parameter	Parameter Type	Data Type	Description
ATWarnings	Output	DINT	Warning for the AutoTune sequence. Possible value are: 0 = No auto tune done. 1 = In auto tune mode. 2 = Auto tune done. -1 = Error 1: Input automatically set to TRUE, no auto tune possible. -2 = Error 2: Auto tune error, the ATDynamSet expired.
OutGains	Output	GAIN_PID	Gains calculated from AutoTune Sequences. Use GAIN_PID data type to define the OutGains output.
ENO	Output	BOOL	Enable output. Applicable to Ladder Diagram programs.

GAIN_PID Data Type

Parameter	Type	Description
DirectActing	BOOL	Types of acting: TRUE = Direct acting, output moves same direction as error. That is, the actual process value is greater than the SetPoint and the appropriate controller action is to increase the output. For example, Chilling. FALSE = Reverse acting, output moves opposite direction as error. That is, the actual process value is greater than the Setpoint and the appropriate controller action is to decrease the output. For example, Heating.
ProportionalGain	REAL	Proportional gain for PID (>= 0.0001). **Proportional gain for PID (P_Gain)** A higher proportional gain causes a larger change in the output based upon the difference between the PV (measured process value) and SV (set point value). The higher the gain, the faster the error is decreased, but this may result in instability such as oscillations. The lower the gain, the slower the error is decreased, but the system is more stable and less sensitive to large errors. The P_Gain usually is the most important gain to adjust and the first gain to adjust while tuning.

GAIN_PID Data Type

Parameter	Type	Description
TimeIntegral	REAL	Time integral value for PID (>= 0.0001). **Time integral value for PID** A smaller integral time constant causes a faster change in the output based upon the difference between the PV (measured process value) and SV (set point value) integrated over this time. A smaller integral time constant decreases the steady state error (error when SV is not being changed) but increases the chances of instability such as oscillations. A larger integral time constant slows down the response of the system and make it more stable, but PV approaches the SV at a slower rate.
TimeDerivative	REAL	Time derivative value for PID (> 0.0). **Time derivative value for PID (Td)** A smaller derivative time constant causes a faster change in the output based upon the rate of change of the difference between PV (measured process value) and SV (set point value). A smaller derivative time constant makes a system more responsive to sudden changes in error (SV is changed) but increases the chances of instability such as oscillations. A larger time constant makes a system less responsive to sudden changes in error and the system is less susceptible to noise and step changes in PV. TimeDerivative (Td) is related to the derivative gain but allows the derivative contribution to PID to be tuned using time so the sample time must be taken into consideration.
DerivativeGain	REAL	Derivative gain for PID (>= 0.0). **Derivative gain for PID (D_Gain)** A higher derivative gain causes a larger change in the output based upon the rate of change of the difference between the PV (measured process value) and SV (set point value). A higher gain makes a system more responsive to sudden changes in error but increases the chances of instability such as oscillations. A lower gain makes a system less responsive to sudden changes in error and makes the system less susceptible to noise and step changes in the PV. **If derivative gain is set to zero, it disables the derivative portion of the PID.**

AT_Param Data Type

Parameter	Type	Description
Load	REAL	Load parameter for auto tuning. This is the output value when starting AutoTune.
Deviation	REAL	Deviation for auto tuning. This is the standard deviation used to evaluate the noise band needed for AutoTune (noise band = 3* Deviation)(1)
Step	REAL	Step value for AutoTune. Must be greater than noise band and less than ½ load.
ATDynamSet	REAL	Waiting time in seconds before abandoning auto tune.
ATReset	BOOL	Determines whether the output value is reset to zero after an AutoTune sequence: TRUE = Reset output to zero. FALSE = Leaves output at Load value.

(1) The application engineer can estimate the value of ATParams.Deviation by observing the value of Proces input. For example, in a project that involves the control of temperature, if the temperature stabilizes around 22 °C, and a fluctuation of 21.7…22.5 °C is observed, the value of ATParams.Deviation will be (22.5-21.7)/2=0.4.

How to Autotune

Before you autotune, you need to:

- Verify that your system is constant when there is no control. For example, for temperature control, process value should remain at room temperature when there is no control output.
- Configure the set point to 0.
- Set Auto Input to False.
- Set the Gain parameter as follows:

GAIN Parameter Values

GAIN Parameter	Value
DirectActing	According to operation: TRUE (for example, Cooling), or FALSE (for example, Heating)
DerivativeGain	0.5
ProportionalGain	0.0001
TimeIntegral	0.0001
TimeDerivative	0.0

- Set the AT_Parameter as follows:

AT_Parameter Values

AT Parameter	Recommendation
Load	Every 'Load' provides a saturated process value over a period of time. Adjust the load to the value for the saturated process value you want. **IMPORTANT**: If a load of 40 gives you a process value of 30 °C over a period of time, and you want to tune your system to 30 °C, you should set the load to 40.
Deviation	This parameter plays a significant role in the autotune process. The method of deriving this value is explained later in this section. It is not necessary to set this parameter prior to autotuning. However, if you already know the deviation, it is fine to set it first.
Step	Step value should be between 3*Deviation and ½ load. The step provides an offset for the load during autotuning. It should be set to a value high enough to create a significant change in process value.
ATDynamSet	Set this value to a reasonably long time for the autotune process. Every system is different, so allow more time to a system with a process value that takes longer to react to change.
ATReset	Set this parameter to TRUE to reset the output to zero after the autotune process completes. Set this parameter to FALSE to leave the output at load value after the autotune process completes.

During autotune, the controller will automatically set the process value to zero. To autotune, perform the following steps:

1. Set the Initialize input to TRUE.
2. Set the AutoTune input to TRUE.
3. Wait for the Process input to stabilize or reach a steady state.

4. Note the temperature fluctuation of the process value.
5. Calculate deviation value with reference to the fluctuation. For example, if the temperature stabilizes around 22 °C (72 °F) with a fluctuation of 21.7...22.5 °C (71...72.5 °F), the value of 'ATParams.Deviation' is:

$$\text{For °C: } \frac{22.5 - 21.7}{2} = 0.4 \qquad \text{For °F: } \frac{72.5 - 71}{2} = 0.75$$

6. Set the deviation value, if you have not set it yet.
7. Change the initialize input to FALSE.
8. Wait until the 'AT_Warning' shows 2. The autotune process is successful.
9. Get the tuned value from the 'OutGains'.

How Autotune Works

The auto tune process begins when the 'Initialize' is set to FALSE (Step 7.) At this moment, the control output increases by the amount of 'Step' and the process waits for the process value to reach or exceeds 'first peak'.

First peak is defined as:

For Direct Operation: First peak = PV1 - (12 x Deviation)
For Reverse Operation: First peak = PV1 + (12 x Deviation)
Where PV1 is the process value when Initialize is set to FALSE.

Once the process value reaches first peak, the control output reduces by the amount of Step and waits for the process value to drop to the second peak.

Second peak is defined as:

For Direct Operation: Second peak = PV1 - (3 x Deviation)
For Reverse Operation: Second peak = PV1 + (3 x Deviation)
Once the process value reaches or falls below second peak, calculations commence and a set of gain will be generated to parameter OutGains.

Troubleshooting an Autotune Process

You can tell what is going on behind the autotune process from the sequences of control output. Here are some known sequences of control output and what it means if autotune fails. For the ease of illustrating the sequence of control output, we define:
Load: 50
Step: 20

Output Sequence 1: 50 -> 70 -> 30

Sequence Condition	Autotune Result	Action for Autotune Fail
Process value reached 'first peak' and 'second' peak in time	Likely successful	NA

Output Sequence 2: 50 -> 70 -> 50

Sequence Condition	Autotune Result	Action for Autotune Fail
Process value not able to reach 'first peak'	Likely unsuccessful	Reduce Deviation or Increase Step

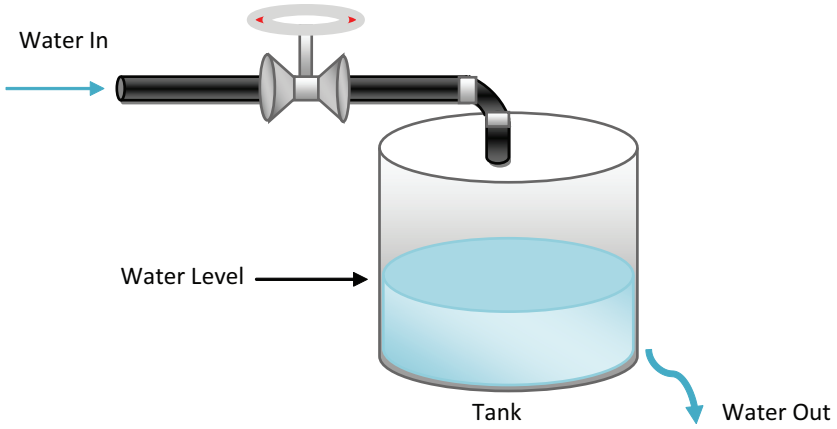
Output Sequence 3: 50 -> 70 -> 30 -> 50

Sequence Condition	Autotune Result	Action for Autotune Fail
Process value not able to reach second peak	Likely unsuccessful	Increase Deviation or increase Step

Output Sequence 4: 50 -> 70

Sequence Condition	Autotune Result	Action for Autotune Fail
Process value not able to reach First peak in time	Likely unsuccessful	Increase ATDynamSet

PID Application Example



The illustration above shows a basic water level control system, to maintain a preset water level in the tank. A solenoid valve is used to control incoming water, filling the tank at a preset rate. Similarly, outflowing water is controlled at a measureable rate.

IPID Autotuning for First and Second Order Systems

Autotune of IPID can only work on first and second order systems.

A first order system can be described by a single independent energy storage element. Examples of first order systems are the cooling of a fluid tank, the flow of fluid from a tank, a motor with constant torque driving a disk flywheel or an

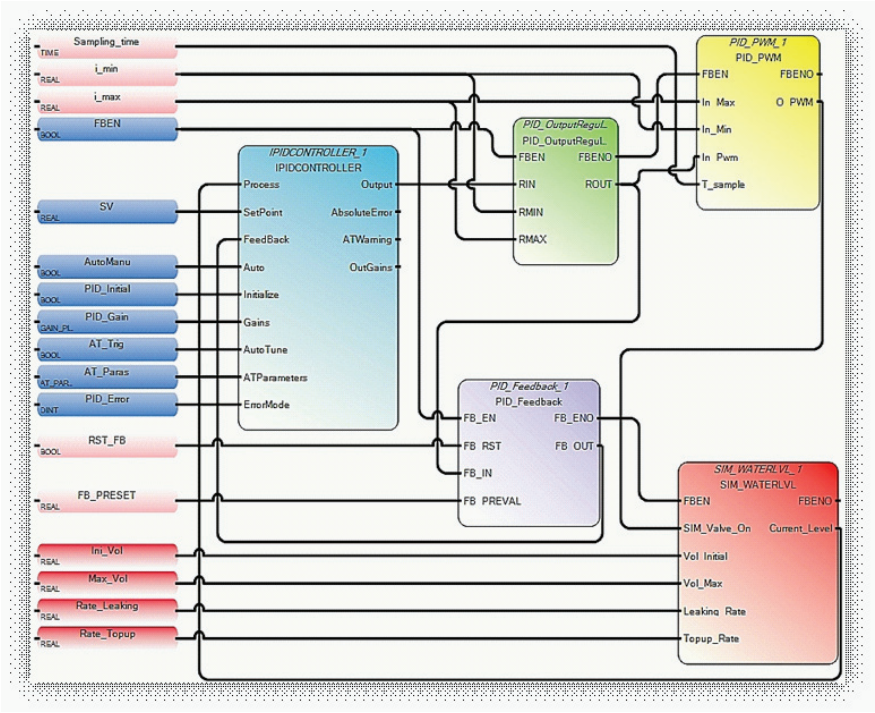
electric RC lead network. The energy storage element for these systems are heat energy, potential energy, rotational kinetic energy and capacitive storage energy, respectively.

This may be written in a standard form such as $f(t) = \tau dy/dt + y(t)$, where τ is the system time constant, f is the forcing function and y is the system state variable.

In the cooling of a fluid tank example, it can be modeled by the thermal capacitance C of the fluid and thermal resistance R of the walls of the tank. The system time constant will be RC, the forcing function will be the ambient temperature and the system state variable will be the fluid temperature.

A second order system can be described by two independent energy storage elements which exchange stored energy. Examples of second order systems are a motor driving a disk flywheel with the motor coupled to the flywheel via a shaft with torsional stiffness or an electric circuit composed of a current source driving a series LR (inductor and resistor) with a shunt C (capacitor). The energy storage elements for these systems are the rotational kinetic energy and torsion spring energy for the former and the inductive and capacitive storage energy for the latter. Motor drive systems and heating systems can be typically modeled by the LR and C electric circuit.

PID Code Sample



The illustration PID Code Sample shows sample code for controlling the PID application example shown before. Developed using Function Block Diagrams, it

consists of a pre-defined function block, IPIDCONTROLLER, and four use-defined function blocks. These four are:

- PID_OutputRegulator
  This user-defined function block regulates the output of IPIDCONTROLLER within a safe range to ensure that there is no damage to the hardware used in the process.

  IF RMIN ≤ RIN ≤ RMAX, then ROUT = RIN,
  IF RIN < RMIN, then ROUT = RMIN,
  IF RIN > RMAX, then ROUT = RMAX.

- PID_Feedback
  This user defined function block acts as a multiplexer.

  IF "FB_RST" is false, FB_OUT=FB_IN;
  If "FB_RST" is true, then FB_OUT=FB_PREVAL.

- PID_PWM
  This user defined function block provides a PWM function, converting a real value to a time related ON/OFF output.
- SIM_WATERLVL
  This user defined function block simulates the process depicted in the application example shown before.

IMPORTANT User Program Scan Time is Important

The autotuning method needs to cause the output of the control loop to oscillate. In order to identify the oscillation period, the IPID must be called frequently enough to be able to sample the oscillation adequately. The scan time of the user program must be less than half the oscillation period. In essence the Shannon, or Nyquist-Shannon, or the sampling theorem must be adhered to.

In addition, it is important that the function block is executed at a relatively constant time interval. One can typically achieve this using STI interrupt.

System Loading

Micro830 and Micro850 Power Requirements

Controller/Module	Power Requirement
Micro830 and Micro850 (without plug-in/expansion I/O) 10/16-point 24-point 48-point	 5 W 8 W 11 W
Plug-in modules, each	1.44 W
Expansion I/O (system bus power consumption)	2085-IQ16 – 0.85 W 2085-IQ32T – 0.95 W 2085-IA8 – 0.75 W 2085-IM8 – 0.75 W 2085-OA8 – 0.90 W 2085-OB16 – 1.00 W 2085-OV16 – 1.00 W 2085-OW8 – 1.80 W 2085-OW16 – 3.20 W 2085-IF4 – 1.70 W 2085-IF8 – 1.75 W 2085-OF4 – 3.70 W 2085-IRT4 – 2.00 W

Calculate Total Power for Your Micro830/Micro850 Controller

To calculate Total Power for your Micro830 and Micro850 controller, use the following formula:

> **Total Power** =Main Unit Power + No. of Plug-ins * Plug-in Power + Sum of Expansion I/O Power

Example 1:
Derive Total Power for a 24-point Micro830 controller with two plug-ins.

> **Total Power** = 8 W+ 1.44 W * 2 + 0 = **10.88 W**

Example 2:
Derive Total Power for a 48-point Micro850 controller, with 3 plug-ins, and 2085-IQ16 and 2085-IF4 expansion I/O modules attached.

> **Total Power** = 11 W + 3*1.44 W + 0.85 W + 1.7 W = **17.87 W**

Calculate External AC Power Supply Loading for your Micro830 Controller

To calculate External AC Power Supply Loading:

- Get total sensor current loading. For this example, assume it is 250 mA.
- Calculate Total Power Loading by Sensor using this formula: (24V * 250 mA) 6 W.
- Derive External AC Power Supply Loading using this formula: **AC Power Supply Loading** = Total Power calculated for a Micro800 system with Plug in + Total power loading by Sensor

As an example, a 48-point Micro850 controller with2 plug-ins, and 2085-IQ16 and 2085-IF4 expansion I/O, and 250mA sensor current (6W sensor power) will have the following Total Loading for AC Power Supply:

Total loading for AC power supply = 17.87W + 6W = **23.87 W**

ATTENTION: Maximum loading to AC Power Supply is limited to 38.4 W with maximum surrounding ambient temperature limited to 65 °C.

Symbols

__SYSVA_CYCLECNT 70
__SYSVA_TCYCURRENT 70
__SYSVA_TCYMAXIMUM 70

Numerics

1761-CBL-PM02 57
2080-PS120-240VAC 33
2711P-CBL-EX04 7

A

About Your Controller 9
absolute home switch 78, 79
Additional Resources iii
analog cable grounding 48
analog channel wiring guidelines 47
analog inputs
  analog channel wiring guidelines 47
ASCII 51, 53, 58
  configuration 62
AutoTune 274
axis 77
axis output
  general rules 84
axis state diagram 92
axis state update 93
axis states 93

B

before calling for assistance 265

C

cables
  programming 6
  serial port 7
calling for assistance 265
CE mark 19, 20
certifications 19
Checking if Forces (locks) are Enabled 227
CIP Client Messaging 55
CIP communications pass-thru 56
CIP Serial 58
  Parameters 60
CIP Serial Client/Server 51, 53
CIP Serial Driver
  configure 59
  parameters 60
CIP Symbolic Addressing 54
CIP Symbolic Client/Server 51, 54
communication connections 51
communication protocols 51
communications
  ports 51
Compliance to European Union Directive
  EMC Directive 20
  Low Voltage Directive 20
Compliance to European Union Directives 19
Connected Components Workbench v, 9, 69, 73, 93, 163, 166
connection limits 52
ControlFlash 166
controller
  description 3
  grounding 43
  I/O wiring 46
  minimizing electrical noise 47
  preventing excessive heat 26
Controller Error Recovery Model 264
controller load 71
Controller Mounting Dimensions 31
controller password 161
  recover 166
controller security 161

D

deceleration 83
DF1 point-to-point connection 57
DHCP Client 51
DIN Rail Mounting 33
DIN rail mounting 33
direction input 83
disconnecting main power 23

E

EII Function Configuration 250
EII function file 249
EII Function Status Information 251
Embedded Serial Port Cables 7
Embedded Serial Port Wiring 49
EMC Directive 20
enable and valid status
  general rules 86
encoder
  quadrature 138
Endian Configuration 195
error 86
error codes 255, 256

error handling
general rules 86
error recovery model 264
ErrorStop 92
Establishing Communications Between RSLinx and a Micro830 via USB 204
Ethernet
configuration settings 64
EtherNet/IP Client/Server 51
European Union Directive compliance 19
EMC Directive 20
Event Input Interrupt (EII) Function Configuration and Status 250
event input interrupt (EII) function file 249
exclusive access 161
Execution Rules 70

F

fault routine
description of operation 240
operation in relation to main control program 237
priority of interrupts 239
faults
recoverable and non-recoverable 240
force status 254
Forcing I/Os 227

G

general considerations 20
grounding the controller 43
Guidelines and Limitations for Advanced Users 73

H

Hardware Features 1
Hardware Overview 1
heat protection 26
High-Speed Counter (HSC) 128
high-speed counter function file 151
High-Speed Counter Overview 127
home marker 78
housekeeping 69
HSC (High Speed Counter) Function Block 151, 249
HSC APP Data Structure 133
HSC function file 151
HSC Interrupt Configuration 157
HSC Interrupt POU 158
HSC Interrupt Status Information 159
HSC Interrupts 156
HSC STS Data Structure 144
HSC_SET_STS Function Block 153

I

Information About Using Interrupts 237
in-position signal 79
input parameters 83
input states on power down 26
Installation Considerations 20
Installing Your Controller 31
INT instruction 241, 242
interrupt subroutine instruction 241, 242
interrupts
interrupt instructions 241
overview 237
selectable timed start (STS) instruction 241
user fault routine 240
user interrupt disable (UID) instruction 242
user interrupt enable (UIE) instruction 244
user interrupt flush (UIF) instruction 245
IP address
exclusions 66
rules 66
IPIDCONTROLLER 271
parameters 268, 269, 271
isolation transformers
power considerations 25

J

jerk inputs
general rules 83

K

keyswitch 166

L

lower (Negative) Limit switch 78
lower (negative) limit switch 79

M

Mapping Address Space and supported Data Types 195
master control relay 27
emergency-stop switches 28
using ANSI/CSA symbols schematic 30
using IEC symbols schematic 29
master control relay circuit
periodic tests 24
MC_AbortTrigger 81

MC_Halt 82, 87, 89, 91
MC_Home 82
MC_MoveAbsolute 82, 87
MC_MoveRelative 82, 87
MC_MoveVelocity 82, 87
MC_Power 81
MC_ReadAxisError 81
MC_ReadBoolParameter 81
MC_ReadParameter 81
MC_ReadStatus 81
MC_Reset 81, 92
MC_SetPosition 81
MC_Stop 82, 87, 91
MC_TouchProbe 81
MC_WriteBoolParameter 81
MC_WriteParameter 81
Micro800 cycle or scan 69
Micro830 Controllers 2
Micro830 controllers
  inputs/outputs types 5
Micro850 controllers
  inputs/outputs types 6
minimizing electrical noise 47
minimizing electrical noise on analog channels 47
Modbus Mapping 195
Modbus Mapping for Micro800 195
Modbus RTU 51, 53, 58
  configuration 61
Modbus TCP Client/Server 51, 53
Modbus/TCP server 53
Module Spacing 32
motion control 75, 76
  administrative function blocks 81
  general rules 83
  wiring input/output 79
motion control function blocks 81
motion function blocks 77
motor starters (bulletin 509)
  surge suppressors 42
mounting dimensions 31

N

network status 254
Normal Operation 254
North American Hazardous Location Approval 23

O

output active
  general rules 86
output exclusivity 84
output status 254
Overview of Program Execution 69

P

panel mounting 34
  dimensions 34
Performance, MSG_MODBUS 200
PID 268
PID Application Example 276
PID Code Sample 277
PID Function Blocks 267
PLS Data structure 154
PLS Example 155
PLS Operation 154
position/distance input 83
POU (Program Organizational Unit) 70
power considerations
  input states on power down 26
  isolation transformers 25
  loss of power source 25
  other line conditions 26
  overview 25
  power supply inrush 25
power distribution 24
power source
  loss of 25
power status 253
power supply inrush
  power considerations 25
preventing excessive heat 26
Priority of User Interrupts 238
program scan cycle 71
programmable limit switch 127
Programmable Limit Switch (PLS) Function 153
Programmable Limit Switch Overview 127
PTO 75
  configurable input/output 78
  fixed input/output signals 78
PTO direction 78, 79
PTO pulse 78, 79

Q

quadrature encoder 138
Quickstarts 201

R

relative move versus absolute move
general rules 86
RJ-45 Ethernet port 51
RJ-45 ethernet port 7
RS-232/485 combo port 51
RS-232/485 serial port 51
Run Mode Change (RMC) 10
benefits 10
limitations 14
RMC memory 12
uncommitted changes 11
using 229
Run Mode Configuration Change (RMCC) 15
loop-back message 15
using EtherNet/IP 17
using Modbus RTU 16
verify IP address change 19
verify node address change 17

S

safety circuits 24
Safety Considerations 22
safety considerations 22
disconnecting main power 23
hazardous location 23
master control relay circuit
periodic tests 24
periodic tests of master control relay circuit 24
power distribution 24
safety circuits 24
Selectable Time Interrupt (STI) Function Configuration and Status 247
selectable timed start instruction 241
serial communications status 254
serial port
configure 58
servo drive 75
servo/drive on 78, 79
servo/drive ready 78, 79
Shutdown 58
Sockets Client/Server 52, 55
Specifications
Micro800 Programmable Controller External AC Power Supply 193
Micro830 10 Point Controllers 167
Micro830 16 Point Controllers 171
Micro830 24 Point Controllers 174
Micro830 48 Point Controllers 178
Micro830 Relay Charts 183
status indicator 2
ethernet 7
fault status 254
input status 253
module status 7, 254
network status 7, 254
output status 254
power status 253
run status 253
serial communications 254
Status Indicators on the Controller 253
STI Function Configuration 248
STI Function Status Information 248
STS instruction 241
surge suppressors
for motor starters 42
recommended 42
using 40
system assembly
Micro830 and Micro850 24-point controllers 37

T

timing diagrams
quadrature encoder 138
touch probe input switch 78, 79
troubleshooting 253

U

UID instruction 242
UIE instruction 244
UIF instruction 245
upper (Positive) Limit switch 78
upper (positive) limit switch 79
User Defined Function (UDF) 69, 73
User Defined Function Block (UDFB) 69, 73
user fault routine
creating a user fault routine 240
recoverable and non-recoverable faults 240
User Interrupt Configuration 240
user interrupt disable instruction 242
user interrupt enable instruction 244
user interrupt flush instruction 245
using emergency-stop switches 28
Using Interrupts 237
Using the High-Speed Counter and Programmable Limit Switch 127
Using the Selectable Timed Interrupt (STI) Function 247

V

validate IP address 66
variable retainment 73
velocity input 83

W

wiring diagrams 43
Wiring Examples 48
wiring recommendation 39
Wiring Your Controller 39

Notes:

Rockwell Automation Support

Rockwell Automation provides technical information on the Web to assist you in using its products.
At http://www.rockwellautomation.com/support/, you can find technical manuals, a knowledge base of FAQs, technical and application notes, sample code and links to software service packs, and a MySupport feature that you can customize to make the best use of these tools.

For an additional level of technical phone support for installation, configuration, and troubleshooting, we offer TechConnect support programs. For more information, contact your local distributor or Rockwell Automation representative, or visit http://www.rockwellautomation.com/support/.

Installation Assistance

If you experience a problem within the first 24 hours of installation, review the information that is contained in this manual. You can contact Customer Support for initial help in getting your product up and running.

United States or Canada	1.440.646.3434
Outside United States or Canada	Use the Worldwide Locator at http://www.rockwellautomation.com/support/americas/phone_en.html, or contact your local Rockwell Automation representative.

New Product Satisfaction Return

Rockwell Automation tests all of its products to ensure that they are fully operational when shipped from the manufacturing facility. However, if your product is not functioning and needs to be returned, follow these procedures.

United States	Contact your distributor. You must provide a Customer Support case number (call the phone number above to obtain one) to your distributor to complete the return process.
Outside United States	Please contact your local Rockwell Automation representative for the return procedure.

Documentation Feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete this form, publication RA-DU002, available at http://www.rockwellautomation.com/literature/.

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Rockwell Automation Publication 2080-UM002I-EN-E - November 2016
Supersedes Publication 2080-UM002H-EN-E - November 2015

Copyright © 2016 Rockwell Automation, Inc. All rights reserved. Printed in the U.S.A.